Volume 20 Number 3

ACTA CYBERNETICA

Szeged, 2012

ACTA CYBERNETICA

Information for authors. Acta Cybernetica publishes only original papers in the field of Computer Science. Manuscripts must be written in good English. Contributions are accepted for review with the understanding that the same work has not been published elsewhere. Papers previously published in conference proceedings, digests, preprints are eligible for consideration provided that the author informs the Editor at the time of submission and that the papers have undergone substantial revision. If authors have used their own previously published material as a basis for a new submission, they are required to cite the previous work(s) and very clearly indicate how the new submission offers substantively novel or different contributions beyond those of the previously published work(s). Each submission is peer-reviewed by at least two referees. The length of the review process depends on many factors such as the availability of an Editor and the time it takes to locate qualified reviewers. Usually, a review process takes 6 months to be completed. There are no page charges. An electronic version of the puplished paper is provided for the authors in PDF format.

Manuscript Formatting Requirements. All submissions must include a title page with the following elements:

- title of the paper
- author name(s) and affiliation
- name, address and email of the corresponding author
- An abstract clearly stating the nature and significance of the paper. Abstracts must not include mathematical expressions or bibliographic references.

References should appear in a separate bibliography at the end of the paper, with items in alphabetical order referred to by numerals in square brackets. Please prepare your submission as one single PostScript or PDF file including all elements of the manuscript (title page, main text, illustrations, bibliography, etc.). Manuscripts must be submitted by email as a single attachment to either the most competent Editor, the Managing Editor, or the Editor-in-Chief. In addition, your email has to contain the information appearing on the title page as plain ASCII text. When your paper is accepted for publication, you will be asked to send the complete electronic version of your manuscript to the Managing Editor. For technical reasons we can only accept files in LaTeX format.

Subscription Information. Acta Cybernetica is published by the Institute of Informatics, University of Szeged, Hungary. Each volume consists of four issues, two issues are published in a calendar year. Subscription rates for one issue are as follows: 5000 Ft within Hungary, €40 outside Hungary. Special rates for distributors and bulk orders are available upon request from the publisher. Printed issues are delivered by surface mail in Europe, and by air mail to overseas countries. Claims for missing issues are accepted within six months from the publication date. Please address all requests to:

Acta Cybernetica, Institute of Informatics, University of Szeged
P.O. Box 652, H-6701 Szeged, Hungary
Tel: +36 62 546 396, Fax: +36 62 546 397, Email: acta@inf.u-szeged.hu

Web access. The above informations along with the contents of past issues are available at the Acta Cybernetica homepage http://www.inf.u-szeged.hu/actacybernetica/ .

EDITORIAL BOARD

Ferenc Gécseg
Department of Computer Algorithms
and Artificial Intelligence
University of Szeged
Szeged, Hungary
gecseg@inf.u-szeged.hu

Jozef Gruska
Institute of Informatics/Mathematics
Slovak Academy of Science
Bratislava, Slovakia
gruska@savba.sk

Tibor Gyimóthy
Department of Software Engineering
University of Szeged
Szeged, Hungary
gyimothy@inf.u-szeged.hu

Helmut Jürgensen
Department of Computer Science
Middlesex College
The University of Western Ontario
London, Canada
helmut@csd.uwo.ca

Zoltan Kato
Department of Image Processing
and Computer Graphics
Szeged, Hungary
kato@inf.u-szeged.hu

Alice Kelemenová
Institute of Computer Science
Silesian University at Opava
Opava, Czech Republic
Alica.Kelemenova@fpf.slu.cz

László Lovász
Department of Computer Science
Eötvös Loránd University
Budapest, Hungary
lovasz@cs.elte.hu

Gheorghe Păun
Institute of Mathematics of the
Romanian Academy
Bucharest, Romania
George.Paun@imar.ro

András Prékopa
Department of Operations Research
Eötvös Loránd University
Budapest, Hungary
prekopa@cs.elte.hu

Arto Salomaa
Department of Mathematics
University of Turku
Turku, Finland
asalomaa@utu.fi

László Varga
Department of Software Technology
and Methodology
Eötvös Loránd University
Budapest, Hungary
varga@ludens.elte.hu

Heiko Vogler
Department of Computer Science
Dresden University of Technology
Dresden, Germany
Heiko.Vogler@tu-dresden.de

Gerhard J. Woeginger
Department of Mathematics and
Computer Science
Eindhoven University of Technology
Eindhoven, The Netherlands
gwoegi@win.tue.nl

Acta Cybernetica **20** (2012) 347–384.

Improving the Construction of the DBM Over Approximation of the State Space of Real-time Preemptive Systems*

Abdelli Abdelkrim†

Abstract

We present in this paper an algorithm allowing an efficient computation of the tightest DBM over-approximation of the state space of preemptive systems modeled by using Time Petri Nets with inhibitor arcs. First of all, we propose an algorithm that reduces the effort of computing the tightest DBM over-approximated graph. For this effect, each class of this graph is expressed as a pair $(M, \widetilde{D})$, where M is a marking and $\widetilde{D}$ is the system of all DBM inequalities even the redundant ones. We thereby make it possible to compute the system $\widetilde{D}$ straightforwardly in its normal form, without requiring to compute the intermediary polyhedra. Hence, we succeed to remove the errors reported in the implementation of other DBM approximations. Then we show that by relaxing a bit in the precision of the DBM approximation, we can achieve to construct more compact graphs while reducing still more the cost of their computation. We provide for this abstraction a suitable equivalence relation that contract yet more the graphs. The experimental results comparing the defined constructions with other approaches are reported.

Keywords: Preemptive Systems, Time Petri Nets, Stopwatch Inhibitor arcs, State class graph, DBM over-approximation

1 Introduction

Nowadays, real-time systems are becoming more and more complex and are often critical. Therefore, their verification has to be performed thoroughly in order to prove the correctness of their behaviors. These systems consist of several tasks that are interacting and sharing one or more resources (e.g processors, memory). Hence, the problem is to determine, for instance, whether these actions can be scheduled in such a way that their constraints are satisfied.

Furthermore, the correctness proofs of such systems are demanding much theory regarding their increasing complexity. We may need, for instance, to consider formal

*This work was supported by the Algerian national Project PNR number 8/u160/3067

†LSI Laboratory- Computer Science Department- USTHB University-Algiers Algeria

models requiring the specification of time preemption; concept where execution of a task may be stopped for a while and later resumed at the same point. This notion of suspension requires to extend the semantics of timed clocks in order to handle such behaviors. For this effect, Cassez *et al* have introduced the *stopwatch* mechanism [10] and hence many models have been defined, as for instance, hybrid automata (LHA) [2] and stopwatch automata (SWA) [10]. Time Petri nets (TPN) have also been considered in several works including *Preemptive-TPN* [7], *Stopwatch-TPN* [5], *Inhibitor-TPN* [15], and *Scheduling-TPN* [13].

The verification of qualitative and quantitative properties of such a system on its formal description involves the investigation of a part of or the whole set of its reachable states that determines its state space. As the state space is generally infinite due to dense time semantics, we need therefore to compute finite abstractions of it, that preserve properties of interest. In these abstractions, states are grouped together, in order to obtain a finite number of these groups. These groups of states are, for instance, regions and zones for timed automata, or state classes[4] for time Petri nets. Hence, the states pertaining to each group can be described by a system of linear inequalities, noted D, whose set of solutions determines the state space of the group. Hence, if the model does not use any stopwatch, then D is of a particular form, called *DBM (Difference Bound Matrix) [8]*. However, when using stopwatches, the system D becomes more complex and does not fit anymore into a DBM. In actual fact, D takes a general polyhedral form whose *canonical* form [1] is given as a conjunction of two subsystems $D = \widetilde{D} \wedge \widehat{D}$, where $\widetilde{D}$ is a DBM system and $\widehat{D}$ is a polyhedral system that cannot be encoded with *DBMs*.

The major shortcoming of manipulating polyhedra is the performance loss in terms of computation speed and memory usage. Indeed, the complexity of solving a general polyhedral system is exponential in the worst case, while it is polynomial for a DBM system. Furthermore, the reachability is proved to be undecidable for both SWA and LHA [10] [2][11], as well as for TPN extended with stopwatches [5] even when the net is bounded. As a consequence, the finiteness of the graph cannot be guaranteed.

In order to speed up the state space computation, an idea is to leave out the subsystem $\widehat{D}$, to keep only the system $\widetilde{D}$ approximating thus the space of D to the DBM containing it, see [7][15] for details. The obvious consequence of the over-approximation is that we add states in the computed group that are not reachable indeed. However, this could prevent the graph computation to terminate, by making the number of computed markings unbounded. Conversely, this can also make the computation of the approximated graph terminate by cutting off the polyhedral inequalities that prevent the convergence. It is noteworthy that since the resulted graph encompasses the exact one, only a subset of properties of interest are preserved.

In order to perform efficiently the exact analysis of the over-approximated graph, *Bucci et al* [7] have proposed to use the DBM over-approximation as a pre-computing before cleaning up the graph from its additional sequences that have been added due to over-approximation. This is done by constraining each sequence reachable in the over-approximated graph by a linear system that reproduces the

original timing constraints of the model. Hence, if there is no solution that makes the sequence be firable according to the time constraints of the system, then the sequence has been introduced by the over-approximation and can be cleared up, otherwise the solution set makes it possible to determine the feasible timings of this sequence.

Furthermore, in order to settle a compromise between both techniques, a hybrid approach has been proposed by *Roux et al* [14]. The latter puts forward a sufficient condition that determines the cases where the subsystem $\widehat{D}$ becomes redundant in D. Hence, the combination of both DBM and polyhedral representations makes it possible to build the exact state class graph faster and with lower expenses in terms of memory usage comparatively to the polyhedra based approach [13]. More recently, *Berthomieu et al* have proposed an over-approximation method based on a quantization of the polyhedral system D [5]. The latter approach ends in the exact computation of the graph in almost all cases faster than the hybrid approach [14]. Nevertheless, this technique is more costly in terms of computation time and memory usage comparatively to DBM over-approximation although it yields much precise graphs.

We consider in this paper real time preemptive systems modeled by using $ITPN$ (*Time Petri Nets with inhibitor arcs*) [15]. This model extends TPN with inhibitor arcs to control the progression and the suspension of stopwatches.

First of all, we propose a new algorithm to compute the tightest DBM over-approximation of the state class graph of preemptive systems. For this effect, we express each class, noted $\widetilde{E}$, of the approximated graph as a pair $(M, \widetilde{D})$ where M is a marking and $\widetilde{D}$ is the system of all DBM constraints, even the redundant ones. We show that by maintaining a complete representation of $\widetilde{D}$ and avoiding to compute its minimal form, we achieve to define an efficient algorithm that computes a normalized class in a square time in the number of enabled transitions. Besides, we prove that the systems $\widetilde{D}$ and $\widehat{D}$ are equivalent; this ensures that $\widetilde{D}$ is the tightest DBM over-approximation that one can derive from D. Unlike the other approaches [15][7], our algorithm avoids the computation of the intermediary polyhedra D. We thereby avoid its computation and its manipulation and remove all the costs induced by the derivation of the normal form, even the minimal form of $\widetilde{D}$. This allows to improve significantly the calculation of a class, and to remove the drawbacks that stand in the implementation of other DBM over-approximations.

In the second part of the paper, we propose another abstraction of the state space of an $ITPN$. We show that by relaxing a bit in the precision of the constraints of the system $\widetilde{D}$, we can compute smaller graphs with a minimal cost. However, although this abstraction is less precise than the former, it preserves all the firing sequences of the model and may be sufficient to model check the properties of the $ITPN$. To improve once more this construction, we provide a suitable equivalence relation that contracts the size of the resulted graphs while reducing the effort of their computation. For this effect, we show that for specific transitions, the computation of their firing distances can be useless, and we prove that the equality between these distances is not required in the class equivalence test. This result is

important since it makes it possible to gather in a same node unequal classes that are indeed bisimilar. This contraction leads to an efficient construction of *DBM* approximated graphs that can be in certain cases more appropriate to use to model check the linear properties of the model. Moreover, the experiments show that both constructions are faster in all cases while providing, in general, smaller graphs than other fellow approaches [15][7].

The remainder of this paper is organized as follows: In *Section 2* we present the syntax and the formal semantics of the *ITPN* model. Then, in *Section* 3 we introduce formally our approach. In *Section* 4 we define a new construction of an abstraction of the state space of an *ITPN*. In *Section* 5 we give some experimental results that compare the performances of our algorithms with those of other approaches.

2 Time Petri Net with Inhibitor Arcs

Time Petri nets with inhibitor arcs (*ITPN*) [15] extend time Petri nets [9] with *Stopwatch inhibitor arcs*. Formally, an *ITPN* is defined as follows:

Definition 1. *An ITPN is given by the tuple* (P, T, B, F, M^0, I, IH) *where:* P *and* T *are respectively two nonempty sets of places and transitions;* B *is the backward function* [1] $: B : P \times T \longrightarrow \mathbb{N} = \{0, 1, 2, ..\}$; F *is the forward function* $F : P \times T \longrightarrow \mathbb{N}$; M^0 *is the initial marking mapping* $M^0 : P \longrightarrow \mathbb{N}$; I *is the delay mapping* $I : T \longrightarrow \mathbb{Q}^+ \times \mathbb{Q}^+ \cup \{\infty\}$, *where* $\mathbb{Q}^+$ *is set of non negative rational. We write* $I(t) = [tmin(t), tmax(t)]$ *such that* $0 \leq tmin(t) \leq tmax(t)$; $IH : P \times T \longrightarrow \mathbb{N}$ *is the inhibitor arc function; there is an inhibitor arc connecting the place* p *to the transition* t, *if* $IH(p, t) \neq 0$.

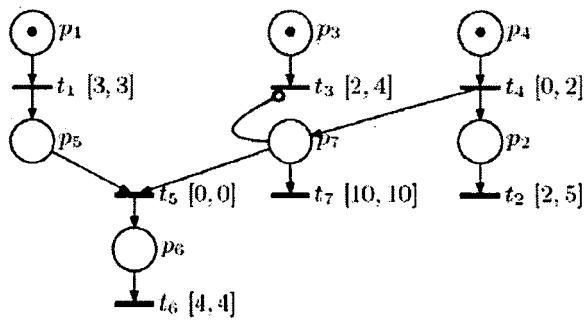

Figure 1: An *ITPN* model

For instance, let us consider the *ITPN* model shown in *Figure 1*, already presented in [15]. Therein, the *inhibitor arc* is the arc ended by a circle that connects

[1] $\mathbb{N}$ denotes the set of positive integers. In the graphical representation, we represent only arcs of non null valuation, and those valued 1 are implicit.

the place p_7 to the transition t_3. Initially, the place p_3 is marked but not the place p_7; hence t_3 is enabled but not inhibited. Therefore, t_3 is progressing, as t_4 which is also enabled for the initial marking. However, the firing of the transition t_4 consumes the token in the place p_4 and produces a new one in p_2 and another one in p_7. Therefore, the inhibitor arc connected to t_3 is activated and hence the clock of t_3 is suspended (t_3 is thus inhibited); this time suspension lasts as long as p_7 remains marked. For more details, the formal semantics of the $ITPN$ model is introduced in the next section.

Let $RT := (P, T, B, F, M^0, I, IH)$ be an $ITPN$.

- We call a *marking* the mapping, noted M, which associates with each place a number of tokens: $M : P \rightarrow \mathbb{N}$.

- A transition t is said to be *enabled* for the marking M, if $\forall p \in P, B(p,t) \leq M(p)$; the number of tokens in each input place of t is greater or equal to the valuation of the arc connecting this place to the transition t. Thereafter, we denote by $Te(M)$ the set of transitions *enabled* for the marking M.

- A transition t is said to be *inhibited* for a marking M, if it is *enabled* and if there exists an inhibitor arc connected to t, such that the marking satisfies its valuation $(t \in Te(M)) \wedge \exists p \in P, 0 < IH(p,t) \leq M(p)$. We denote by $Ti(M)$ the set of transitions that are *inhibited* for the marking M.

- A transition t is said to be *activated* for a marking M, if it is enabled and not inhibited, $(t \in Te(M)) \wedge \ (t \notin Ti(M))$; we denote by $Ta(M)$ the set of transitions that are *activated* for the marking M.

- Let M be a marking ; two transitions t_i and t_j enabled for M are said to be *conflicting* for M, if $\exists p \in P, \quad B(p,t_i) + B(p,t_j) > M(p)$.

- We note $Conf(M)$ the relation built on $Te(M)^2$ such that $(t_1, t_2) \in Conf(M)$, iff t_1 and t_2 are in conflict for the marking M.

For instance, let us consider again the $ITPN$ of *Figure 1*; its initial marking is equal to $M^0 : \{p_1, p_3, p_4\} \rightarrow 1; \{p_2, p_5, p_6, p_7\} \rightarrow 0$; the sets of enabled, inhibited, and activated transitions for M^0 are respectively $Te(M^0) = \{t_1\}$, $Ti(M^0) = \varnothing$, and $Ta(M^0) = Te(M^0)$.

Remark 1. We assume in the sequel a monoserver semantics, which means that for a given marking a transition can be enabled at most once .

We define the semantics of an $ITPN$ as follows:

Definition 2. *The semantics of an $ITPN$ is defined as a LTS (labeled transition system), $ST = (\Gamma, e^0, \rightarrow)$, such that:*

- Γ *is the set of reachable states: Each state, noted* e, *pertaining to* Γ *is a pair* (M, V) *where* M *is a marking and* V *is a valuation function that associates with each enabled transition* t *of* $Te(M)$ *a time interval that gives the range of relative times within which* t *can be fired. Formally we have :* $\forall t \in Te(M), \quad V(t) := [x(t), y(t)]$

- $e^0 = (M^0, V^0)$ *is the initial state, such that:* $\forall t \in Te(M^0), \quad V^0(t) := I(t) := [tmin(t), tmax(t)]$.

- $\rightarrow \in \Gamma \times (T \times \mathbb{Q}^+) \times \Gamma$ *is a transition relation, such that* $((M, V), (t_f, \underline{t_f}), (M^\uparrow, V^\uparrow)) \in \rightarrow$, *iff:*

  (i) $t_f \in Ta(M)$.

  (ii) $x(t_f) \leq \underline{t_f} \leq \underset{\forall t \in Ta(M)}{MIN} \{y(t)\}$.

  and we have:

  $\forall p \in P,\ M^\uparrow(p) := M(p) - B(p, t_f) + F(p, t_f)$.

  $\forall t \in Te(M^\uparrow)$

  if $t \notin New(M^\uparrow)$*:*

  $$[x^\uparrow(t),\ y^\uparrow(t)] := [MAX(0,\ x(t) - \underline{t_f}),\ y(t) - \underline{t_f}] \quad t \in Ta(M)$$
  $$[x^\uparrow(t),\ y^\uparrow(t)] := [x(t),\ y(t)] \quad t \in Ti(M)$$

  if $t \in New(M^\uparrow)$

  $$[x^\uparrow(t),\ y^\uparrow(t)] := I(t) = [tmin(t),\ tmax(t)]$$

  - *where* $New(M^\uparrow)$ *denotes the set of transitions newly enabled for the marking* $M^\uparrow$. *These transitions are those enabled for*$M^\uparrow$ *and not for* M, *or those enabled for* $M^\uparrow$ *and* M *but are conflicting with* t_f *for the marking* M. *Otherwise, an enabled transition which does not belong to* $New(M^\uparrow)$ *is said to be persistent.*

If t is an enabled transition for a state e, we note $\underline{t}$ the clock associated with t that takes its values in $\mathbb{Q}^+$. $\underline{t}$ measures the residual time of the transition t relatively to the instant where the state e is reached. The time progresses only for activated transitions, whereas it is suspended for inhibited transitions. Therefore, a transition t_f can be fired at relative time $\underline{t_f}$ from a reachable state e, if (i) t_f is *activated* for the marking M, and if (ii) the time can progress within the firing interval of t_f while satisfying the time constraints of other activated transitions. After firing t_f the reachable state, noted $e^\uparrow$, is obtained:

- by consuming a number of tokens in each input place p of t_f (given by the value $B(p, t_f)$), and by producing a number of tokens in each output place p of t_f (given by the value $F(p, t_f)$);

- by shifting the interval of a persistent activated transition with the value of the firing time of t_f. However, the intervals of persistent inhibited transitions remain unchanged. Finally, newly enabled transitions are assigned their static firing intervals.

Similarly as for TPN, the behavior of an $ITPN$ can be defined as a timed sequence of pairs (t_f, δ), where t_f is a transition of the net and $\delta \in \mathbb{Q}^+$. Therefore, the timed sequence $S^* = ((t_f^1, \delta^1), (t_f^2, \delta^2), .., (t_f^n, \delta^n))$ denotes that t_f^1 is fired after $\underline{t_f^1}= \delta^1$ time units, then t_f^2 is fired at relative time $\underline{t_f^2}=\delta^2$ and so on, such that t_f^n is fired at relative time $\underline{t_f^n}=\delta^n$ after an absolute time $\sum_{i=1}^{n} \delta^i$. Moreover, we often express the behavior of the net as an *untimed sequence,* denoted by S, obtained from a timed sequence S^t by removing the firing times: If $S^* = ((t_f^1, \delta^1), (t_f^2, \delta^2), ..., (t_f^n, \delta^n))$, then $S = (t_f^1, t_f^2, .., t_f^n)$. Furthermore, a marking M is said to be *reachable* in ST if there exists an untimed sequence S in ST, going from the initial marking M^0 towards M. As the set of time values is assumed to be dense, the model ST is infinite. In order to analyze this model, we need to compute an abstraction of it that saves the most interesting properties. The *symbolic graph construction* [12] preserves the untimed sequences of ST, and makes it possible to compute a finite graph in almost all cases. However, this contraction might be infinite too when the number of reachable markings is unbounded. As this last property is undecidable for $ITPN$[15], there is no guarantee to compute a finite graph. We show hereafter how to compute the state class graph of the $ITPN$ that preserves the linear properties of the model.

3 $ITPN$ state space construction

For a TPN[9], the state class graph method [4] computes a symbolic graph that preserves mainly the linear properties of the model. Similarly, this construction can be applied to an $ITPN$. This consists in regrouping in a same class all the states reachable after firing the same untimed sequence of transitions; all the states of a same class have the same marking M. Hence, a class is defined by the pair (M, D) where M is the common marking of all the states of the class, and D is a set of inequalities encoding the firing space of the class. D is of a general form, normal form of which is expressed as a conjunction of two subsystems $\overrightarrow{D} \wedge \widehat{D}$. Actually, $\overrightarrow{D}$ contains only DBM constraints and $\widehat{D}$ contains all other constraints than DBM. In the sequel, we refer to $\overrightarrow{D}$ as the tightest DBM system that over-approximates the system D. More formally, a class is defined as follows:

Definition 3. *Let $ST = (\Gamma, e^0, \rightarrow)$ be the LTS associated with an $ITPN$. A class of states of an $ITPN$, noted E, is the set of all the states pertaining to Γ that are reachable after firing the same untimed sequence $S = (t_f^1, .., t_f^n)$ from the initial state e^0. A class E is defined by (M, D), where M is the marking reachable after firing S, and D is the firing space encoded as a set of inequalities. For $Te(M) = \{t_1, .., t_s\}$, we have : $D = \widehat{D} \wedge \overrightarrow{D}$*

$$\overrightarrow{D} := \begin{cases} \bigwedge_{i \neq j} & (\underline{t_j} - \underline{t_i} \leq d_{ij}) \\ \bigwedge_{i \leq s} & (d_{i\bullet} \leq \underline{t_i} \leq d_{\bullet i}) \end{cases}$$

with $(t_j, t_i) \in Te(M)^2 \quad d_{ij} \in \mathbb{Q} \cup \{\infty\}, \; d_{\bullet i} \in \mathbb{Q}^+ \cup \{\infty\}, d_{i\bullet} \in \mathbb{Q}^+$

$$\widehat{D} := \bigwedge_{k=1..p} \quad (\alpha_{1k}\underline{t_1} + .. + \alpha_{sk}\underline{t_s} \leq d_k)$$

with $d_k \in \mathbb{Q} \cup \{\infty\}$, $(\alpha_{1k}, .., \alpha_{sk}) \in \mathbb{Z}^s, p \in \mathbb{N}$ *and*[2]

$$\forall k, \exists (i,j), (\alpha_{ik}, \alpha_{jk}) \notin \{(0,0), (0,1), (0,-1), (1,-1)\}$$

We denote by the element $\{\bullet\}$ the instant at which the class E is reached. Therefore, the value of the clock $\underline{t_i}$ expresses the time relative to the instant $\bullet$, at which the transition t_i can be fired. To each valuation ψ satisfying the system D, corresponds a unique state $e = (M, V)$ reachable in ST after firing the sequence S.

In case of a TPN, the system D is reduced to the subsystem $\overrightarrow{D}$. The inequalities of the latter have a particular form, called DBM (*Difference Bound Matrix*)[8]. The coefficients, $d_{\bullet i}, d_{i\bullet}$ and d_{ij} are respectively, the minimum residual time to fire the transition t_i, the maximum residual time to fire the transition t_i, and the maximal firing distance of the transition t_i relatively to t_j. It should be noticed that the value of $d_{\bullet i}$ and $d_{i\bullet}$ are always positive or null, whereas the value of d_{ij} can be negative, thus denoting that there exists no state e reachable in E, such that t_i can be firable from e.

For a TPN, the firing space of a class can be always encoded as a DBM system. This form makes it possible to apply an efficient algorithm to compute the reachable class from a class E. The overall complexity of this algorithm is $O(m^3)$, where m is the number of enabled transitions in E. However, for a TPN augmented with stopwatches, the state space of a class may require polyhedra to be encoded, manipulation of which induces an exponential complexity in the worst case. The exact state class graph of an $ITPN$ is computed by enumerating and exploring all the classes reachable from the initial class E^0. However, as the number of reachable classes may be unbounded, the termination of the algorithm is undecidable. Formally, the exact state class graph of an $ITPN$ can be defined as follows [5]:

Definition 4. *The exact state class graph of an ITPN, denoted by GR, is the tuple* $(CE, E^0, \longmapsto)$ *where:*

- CE *is the set of classes reachable in GR;*
- $E^0 = (M^0, D^0)$ *is the initial class;*
  $D^0 = \{ \; \forall t_i \in Te(M^0), \quad tmin(t_i) \leq \underline{t_i} \leq tmax(t_i) \; \}$
- $\longmapsto$ *is the transition relation between classes defined on* $CE \times T \times CE$, *such that* $((M, D), t_f, (M^{\uparrow}, D^{\uparrow})) \in \longmapsto$, *iff:*

  a) t_f *is activated and the system D augmented with the firing constraints of* t_f *that we write* $D_a = D \wedge (\forall t \in Ta(M), \quad \underline{t_f} \leq \underline{t})$ *holds.*

[2] $\mathbb{Z}$ denotes the set of relative integers.

b) $\forall p \in P,\ M^{\uparrow}(p) := M(p) - B(p,t_f) + F(p,t_f)$.

c) *The system $D^{\uparrow}$ is computed from D, as follows:*

1. *In the system D_a, replace each variable $\underline{t}$ (related to an enabled transition that is not inhibited for M), by: $\underline{t} := \underline{t_f} + \underline{t'}$, thus denoting the time progression.*
2. *Eliminate then by substitution the variable $\underline{t_f}$ as well as all the variables relative to transitions disabled by the firing of t_f;*
3. *Add to the system thus computed, the time constraints relative to each newly enabled transition for $M^{\uparrow}$:*
   $\forall t_i \in New(M^{\uparrow}),\quad tmin(t_i) \leq \underline{t_i} \leq tmax(t_i)$

The last definition shows how the exact state class graph of an $ITPN$ is built. Being given a class $E = (M, D)$ and a transition t_f activated for M, the computation of a class $E^{\uparrow} = (M^{\uparrow}, D^{\uparrow})$ reachable from E by firing t_f consists in computing the reachable marking $M^{\uparrow}$ and the firing space induced by the new system $D^{\uparrow}$. The class E can fire the transition t_f, if there exists a valuation that satisfies D (a state of E), such that t_f can be fired before all the other activated transitions. The firing of t_f produces a new class $E^{\uparrow} = (M^{\uparrow}, D^{\uparrow})$ which gathers all the states reachable from those of E. The system $D^{\uparrow}$ that encodes the space of $E^{\uparrow}$ is computed from the system D augmented with the firing constraints of t_f . The substitution of variables relative to activated transitions allows shifting the time origin to the instant at which the new class $E^{\uparrow}$ is reached. Then, a new system is computed wherein the variables of transitions disabled following the firing of t_f are removed. Finally, the time constraints relative to newly enabled transitions are added.

The complexity of the firing test and the step 2 of the algorithm, depends on the form of the system D. If D includes polyhedral constraints, then the complexity of the algorithm is exponential, otherwise it is polynomial. The initial system D^0 is always in DBM form, and polyhedral constraints may appear in reachable classes only when inhibited and activated transitions are both persistently enabled in a firing sequence [7].

Knowing how to compute the successors of a class, the state class graph computation is based on a depth-first or breadth-first strategy. Then the state class graph is given as the quotient of GR by a suitable equivalence relation. This equivalence relation may be equality: two classes (M, D) and (M, D') given in their minimal form are equal if $D = D'$, or inclusion; in other terms, if $\rceil D \lceil$ denotes the set of solutions for the system D, then we have : $\rceil D \lceil \subseteq \rceil D' \lceil$. It should be noticed that the equality preserves mainly the untimed language of the model, whereas the inclusion preserves the set of reachable markings. In order to speed up the class' equivalence test, the reachable systems are computed in their minimal form; this implies that all redundant inequalities are removed. Moreover, it is proved, as for DBM systems [4], that the minimal form of a polyhedral system is unique [1]. This property is very important as it permits to detect equivalent classes by comparing their minimal form.

The algorithm given in Definition 4 can be applied to a TPN with the particularity that the system D is always encoded in $DBMs$. Besides, Berthomieu *et al* proved that the number of equivalent DBM systems computed for a TPN is finite [3]. This implies that the resulted graph is necessarily finite, if the number of reachable markings is bounded.

For TPN augmented with stopwatches, the DBM over-approximation technique has been proposed as an alternative solution to analyze preemptive real time systems [15][7]. This approach consists in cutting off the inequalities of the subsystem $\widehat{D}$ when they are generated in D. It thereby keeps only those of the subsystem $\overrightarrow{D}$ to represent an over-approximation of the space of D. This solution makes it possible to build an approximated graph with lesser expenses in terms of computation time and memory usage. In addition, the DBM over-approximation ensures that the number of computed DBM systems is always finite, whereas that of polyhedral systems may be infinite. Therefore, we can compute a finite approximated graph when the computation of the exact graph does not terminate. However, the DBM over-approximation may compute an infinity of unreachable markings while the exact construction is indeed bounded. For a better understanding of how this technique works, we apply the state class graph method to the $ITPN$ example of *Figure 1*. Let $E' = (M', D')$ be the class reachable in the exact graph after firing the sequence $S = (t_4, t_1, t_5)$ from the initial class $E^0 = (M^0, D^0)$.

$$E^0 = \begin{pmatrix} M^0 : p_1, p_3, p_4 \rightarrow 1 \\ D^0 : \begin{cases} 3 \leq \underline{t_1} \leq 3 \\ 2 \leq \underline{t_3} \leq 4 \\ 0 \leq \underline{t_4} \leq 2 \end{cases} \end{pmatrix} \qquad E' = \begin{pmatrix} M' : p_1, p_4, p_6 \rightarrow 1 \\ D' : \begin{cases} 0 \leq \underline{t_2} \leq 4 & 4 \leq \underline{t_6} \leq 4 \\ 0 \leq \underline{t_3} \leq 4 & 1 \leq \underline{t_2} + \underline{t_3} \leq 6 \end{cases} \end{pmatrix}$$

We notice that the transition t_6 is not firable from E' since t_2 or t_3 should be fired before. Put in other way, the firing of t_6 requires that the system $D' \wedge (\underline{t_6} \leq \underline{t_3}) \wedge (\underline{t_6} \leq \underline{t_2})$ admits at least one solution; we should check whether $(\underline{t_2} = \underline{t_3} = \underline{t_6} = 4) \wedge (\underline{t_2} + \underline{t_3} \leq 6)$ holds, or not. As this last inequality is not satisfied, therefore t_6 cannot fire. The system D' contains a polyhedral constraint that cannot be reduced to a DBM. The DBM over-approximation consists in cutting off the polyhedron $1 \leq \underline{t_2} + \underline{t_3} \leq 6$ to leave only DBM constraints to represent the state space of E'. However, by doing this, t_6 becomes firable since $\underline{t_6} = 4$ holds. Therefore, the system $\widetilde{D'} = \overrightarrow{D'}$ denotes an over-approximation of the system D'. In other words, we add new states in the class E' that are not reachable indeed. Nevertheless, this construction makes it possible to preserve a subset of properties.

The computation of the tightest DBM over-approximation of a class E can be obtained by applying different algorithms [15][7]. To reduce the memory usage and to ease the equivalence test, the previous algorithms compute the DBM system of each class in its minimal form. These approaches proceed first to compute the polyhedra, then eliminate the non DBM constraints while normalizing the remaining ones. Then the process terminates by computing the final system in its minimal form.

However, the implementation of the algorithm defined [15] in ROMEO [17] has revealed a loss in the precision of the DBM approximation. This is due to some

improvements in the computation of normalized systems. On the other side, the computation of the graphs in ORIS [16] (which implements the approach defined [7]), reports very slow times although the resulted graphs are correct.

We show in the sequel that by maintaining all the DBM constraints, even the redundant ones, we succeed to compute the tightest DBM approximated class in $o(m^2)$. In concrete terms, we show that by avoiding to calculate the minimal form, we succeed to define an algorithm that computes straightforwardly a normalized DBM system. We thereby eliminate the computation and the manipulation of the intermediary polyhedra that stand in the other algorithms. Moreover, we improve greatly the implementation of the graph construction and remove the bugs reported in *ROMEO*.

Formally, a DBM over-approximated class of an $ITPN$ can be defined as follows:

Definition 5. ***(Approximated Class).*** *A DBM over-approximated class of an $ITPN$, noted $\widetilde{E}$, is the pair $(M, \widetilde{D})$ such that : M is a marking and $\widetilde{D}$ is the full system of all DBM normalized inequalities, involving all variables of transitions enabled for M:*

$$\widetilde{D} = \begin{cases} \bigwedge_{\forall (t_i,t_j) \in Te(M)^2} \quad (\underline{t_j} - \underline{t_i} \leq \widetilde{d_{ij}}) \\ \bigwedge_{\forall t_i \in Te(M)} \quad (\widetilde{d_{i\bullet}} \leq \underline{t_i} \leq \widetilde{d_{\bullet i}}) \end{cases}$$

$$\text{with } (t_j \neq t_i),\ \widetilde{d_{ij}} \in \mathbb{Q} \cup \{\infty\},\ \widetilde{d_{\bullet i}} \in \mathbb{Q}^+ \cup \{\infty\}, \widetilde{d_{i\bullet}} \in \mathbb{Q}^+ :$$

such that each inequality is in the normal form:

$$\forall x, y, z \in Te(M),\ \left(\widetilde{d_{xy}} \leq \widetilde{d_{xz}} + \widetilde{d_{zy}}\right) \wedge \left(\widetilde{d_{xy}} \leq \widetilde{d_{\bullet y}} - \widetilde{d_{x\bullet}}\right).$$

The space of a DBM over-approximated class is encoded by the system $\widetilde{D}$. Besides, we assume that the system $\widetilde{D}$ is given in its *normal* form. As for the minimal form[3], it is proved that this form is unique for a DBM system [7] ; all equivalent systems have the same normal form.

In the sequel, we encode the system $\widetilde{D}$ as a square matrix where each line and corresponding column, are indexed by an element of $Te(M) \cup \{\bullet\}$. In concrete terms, we have:

$$\forall (t_i, t_j) \in Te(M)^2 \wedge (t_i \neq t_j),\quad \widetilde{D}[\bullet, t_i] := \widetilde{d_{\bullet i}};\quad \widetilde{D}[t_i, \bullet] := -\widetilde{d_{i\bullet}}\ ;$$
$$\widetilde{D}[t_i, t_j] := \widetilde{d_{ij}}\ ;\quad \widetilde{D}[t_i, t_i] := 0\ ;\qquad \widetilde{D}[\bullet, \bullet] := 0.$$

These matrix notations are used to represent the coefficients of the system $\widetilde{D}$. For example, the matrix shown in *Table 1* encodes the system $\widetilde{D^0} = D^0$ associated with the initial class of the exact graph of the $ITPN$ of *Figure 1.* It is noteworthy that the approximated class $\widetilde{E^0} = (M^0, \widetilde{D^0})$ is in the normal form, and represents an exact over-approximation of the initial class $E^0 = (M^0, D^0)$ of the graph GR. The minimal form of the system $\widetilde{D^0}$ is given by: $(\underline{t_1} = 3) \wedge (2 \leq \underline{t_3} \leq 4) \wedge (0 \leq \underline{t_4} \leq 2)$.

[3]The minimal form of a DBM system is obtained from its normal form by cutting off all redundant inequalities.

Table 1: The matrix representation of the system $\widetilde{D^0}$.

$\widetilde{D^0}$	$\bullet$	t_1	t_3	t_4
$\bullet$	0	3	4	2
t_1	-3	0	1	-1
t_3	-2	1	0	0
t_4	0	3	4	0

Taking on the previous definition, if $E = (M, D)$ is a class reachable in GR, then the class $\widetilde{E} = (M, \widetilde{D})$ is an over-approximation of E, if the space of states of E is included in that of $\widetilde{E}$, and we have: $\rceil D \lceil \subseteq \left\rceil \widetilde{D} \right\lceil$. Hence, by substituting $\widetilde{E}$ for E in the graph GR, it results that the class $\widetilde{E}$ may derive additional sequences that are not firable from E in GR. We thereby obtain an over-approximation of the graph GR, that we build as defined next:

Definition 6. *The graph of DBM over-approximated classes of an ITPN, denoted by $\widetilde{GR}$, is the tuple $(\widetilde{CE}, \widetilde{E^0}, \rightsquigarrow)$, such that :*

- *$\widetilde{CE}$ is the set of approximated classes reachable in $\widetilde{GR}$;*
- *$\widetilde{E^0} = (M^0, \widetilde{D^0}) \in \widetilde{CE}$ is the initial class, such that:*

$$\widetilde{D^0} := \begin{cases} \forall t_i \in Te(M^0), & tmin(t_i) \leq \underline{t_i} \leq tmax(t_i) \\ \forall t_i \neq t_j \in Te(M^0), & \underline{t_j} - \underline{t_i} \leq tmax(t_j) - tmin(t_i) \end{cases}$$

- *$\rightsquigarrow$ is a transition relation between approximated classes defined on $\widetilde{CE} \times T \times \widetilde{CE}$, such that$((M, \widetilde{D}), t_f, (M^\uparrow, \widetilde{D^\uparrow})) \in \rightsquigarrow$, iff :*
  - $(t_f \in Ta(M)) \wedge (\widetilde{\beta}[t_f] \geq 0)$ *such that:* $\forall x \in Te(M) \cup \{\bullet\}$, $\widetilde{\beta}[x] = \underset{\forall t \in Ta(M)}{MIN} \left\{ \widetilde{D}[x, t] \right\}$.
  - $\forall p \in P$, $M^\uparrow(p) := M(p) - B(p, t_f) + F(p, t_f)$.
  - *The coefficients of the DBM inequalities of the system $\widetilde{D^\uparrow}$ are computed from those of $\widetilde{D}$ by applying the following algorithm:*

$\forall t \in Te(M^\uparrow)$

$\widetilde{D^\uparrow}[t, t] := 0; \qquad \widetilde{D^\uparrow}[\bullet, \bullet] := 0;$

If t is persistent

If $t \in Ti(M)$ (t is inhibited for M)

$$\widetilde{D^\uparrow}[t,\bullet] := MIN \begin{pmatrix} \widetilde{D}[t,\bullet] \\ \widetilde{D}[t_f,\bullet] + \widetilde{\beta}[t] \end{pmatrix} \qquad \widetilde{D^\uparrow}[\bullet,t] := MIN \begin{pmatrix} \widetilde{D}[\bullet,t] \\ \widetilde{D}[t_f,t] + \widetilde{\beta}[\bullet] \end{pmatrix}$$

If $t \notin Ti(M)$ *(*t *is not inhibited for* M*)*

$$\widetilde{D^{\uparrow}}[\bullet, t] := \widetilde{D}[t_f, t]\ ; \qquad \widetilde{D^{\uparrow}}[t, \bullet] := \widetilde{\beta}[t].$$

If t *is newly enabled.*

$$\widetilde{D^{\uparrow}}[\bullet, t] := tmax(t)\ ; \qquad \widetilde{D^{\uparrow}}[t, \bullet] := -tmin(t).$$

$\forall (t_1, t_2) \in (Te(M^{\uparrow}))^2 \wedge (t_1 \neq t_2)$

If t_1 *or* t_2 *are newly enabled.*

$$\widetilde{D^{\uparrow}}[t_1, t_2] := \widetilde{D^{\uparrow}}[\bullet, t_2] + \widetilde{D^{\uparrow}}[t_1, \bullet].$$

If t_1 *and* t_2 *are persistent.*

If $(t_1, t_2) \notin (Ti(M))^2$ *(*t_1 *and* t_2 *are not inhibited for* M*)*

$$\widetilde{D^{\uparrow}}[t_1, t_2] := MIN(\widetilde{D}[t_1, t_2], \quad \widetilde{D^{\uparrow}}[\bullet, t_2] + \widetilde{D^{\uparrow}}[t_1, \bullet]).$$

If $(t_1, t_2) \in (Ti(M))^2$ *(*$t_1 t_2$ *are inhibited for* M*)*

$$\widetilde{D^{\uparrow}}[t_1, t_2] := MIN(\widetilde{D}[t_1, t_2], \quad \widetilde{D^{\uparrow}}[\bullet, t_2] + \widetilde{D^{\uparrow}}[t_1, \bullet]).$$

If $(t_1 \in Ti(M)) \wedge (t_2 \notin Ti(M))$ *(Only* t_1 *is inhibited for* M*).*

$$\widetilde{D^{\uparrow}}[t_1, t_2] := MIN(\widetilde{D}[t_1, t_2] + \widetilde{D}[t_f, \bullet], \quad \widetilde{D^{\uparrow}}[\bullet, t_2] + \widetilde{D^{\uparrow}}[t_1, \bullet]).$$

If $(t_1 \notin Ti(M)) \wedge (t_2 \in Ti(M))$ *(Only* t_2 *is inhibited for* M*)*

$$\widetilde{D^{\uparrow}}[t_1, t_2] := MIN(\widetilde{D}[t_1, t_2] + \widetilde{\beta}[\bullet], \quad \widetilde{D^{\uparrow}}[\bullet, t_2] + \widetilde{D^{\uparrow}}[t_1, \bullet]).$$

If t is an activated transition, then $\widetilde{\beta}[t]$ denotes the minimal time distance between its firing time and any other firable transition. Further, $\widetilde{\beta}[\bullet]$ represents the maximal dwelling time in the class $\widetilde{E}$. Therefore, an activated transition t_f is *not firable* from $\widetilde{E}$, if $\widetilde{\beta}[t_f] < 0$. In other words, it does not exist any state reachable in $\widetilde{E}$ such that the valuation of the clock associated with t_f can overtake the minimal bound $tmin(t_f)$. For a better understanding, the *Figure 2.a.* depicts the computation of the coefficients $\widetilde{\beta}[t_a]$, $\widetilde{D^{\uparrow}}[\bullet, t_a]$ and $\widetilde{D^{\uparrow}}[t_a, \bullet]$ for $t_a \in Ta(M)$.

Moreover, we notice that the maximal residual time of an inhibited transition t_h can decrease after firing t_f. Besides, the minimal residual time of t_h can increase. To clarify this point, let us consider the $ITPN$ of *Figure 1.* Initially t_3 is activated with $\widetilde{D}[t_3, \bullet] = -2$, and the model can fire the transition t_4 between $[0, 2]$. After this firing, the place p_7 becomes marked, and t_3 is inhibited for the first time; we have $\widetilde{D}[t_3, \bullet] = 0$. Then, to fire the newly enabled transition t_2, it needs to let time progress at least with $tmin(t_2) = 2$, while the absolute time must not surpass $tmax(t_1) = 3$. This last constraint restricts the state space of the class reachable after firing t_2 only to states that have fire initially t_4 during $[0, 1]$. As a result, the minimal residual time of t_3 increases after the firing of t_2 (see *Figure 2.b)*

In other respects, the firing distance $\widetilde{D}[t_a, t_h]$ between an activated transition t_a and an inhibited transition t_h can only increase after firing t_f, with the maximal

dwelling time[4] in $\widetilde{E}$. Also, the distance $\widetilde{D}[t_h, t_a]$ can only decrease after firing t_f with the the minimal dwelling time[5] in $\widetilde{E}$.

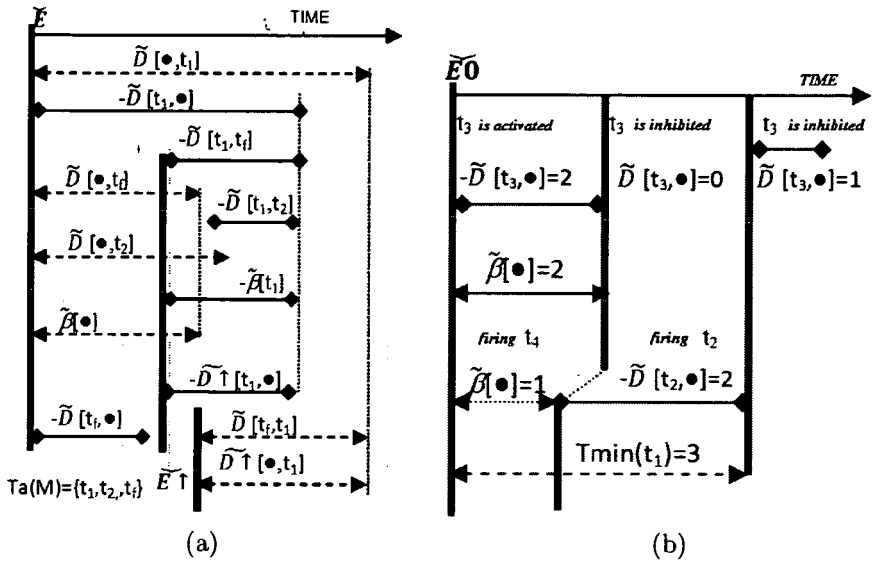

Figure 2: Computing the *DBM* coefficients.

It is noteworthy that if $\widetilde{E}$ is an over-approximation of the exact class E, then all the transitions firable from E are also firable from $\widetilde{E}$. However, a transition which is not firable from E can, on the other hand, be firable[6] from $\widetilde{E}$. Actually, as the class $\widetilde{E}$ contains all the states of E, we can find at least one state e of $\widetilde{E}$ non reachable in E, such that e can fire t_f. We prove hereafter that the algorithm given in Definition 6 computes in all cases the tightest *DBM* over-approximation of the exact graph defined in Definition 4.

Theorem 1. *The graph* $\widetilde{GR} = (\widetilde{CE}, (M^0, \widetilde{D^0}), \rightsquigarrow)$ *is the tightest DBM over-approximation that we can compute from the graph* $GR = (CE, (M^0, D^0), \longmapsto)$.

Proof. We should prove that:

1. $\lceil D^0 \rceil \subseteq \left\lceil \overrightarrow{D^0} \right\rceil = \left\lceil \widetilde{D^0} \right\rceil$.

2. Let be $S = (t_f^1, .., t_f^n)$; if $(M^0, D^0) \xrightarrow{t_f^1} .. \xrightarrow{t_f^n} E = (M, D)$ and $(M^0, \widetilde{D^0}) \overset{t_f^1}{\rightsquigarrow} .. \overset{t_f^n}{\rightsquigarrow} \widetilde{E} = (M, \widetilde{D})$, such that $\lceil D \rceil \subseteq \left\lceil \overrightarrow{D} \right\rceil = \left\lceil \widetilde{D} \right\rceil$; we have, if $E \xrightarrow{t_f} E^{\uparrow} =$

[4]This time denotes the maximal time that has elapsed for t_a and during which t_h has remained suspended.

[5]This time denotes the minimal time that has elapsed for t_a and during which t_h has remained suspended.

[6]Conversely, if t_f is not firable from $\widetilde{E}$, then it is not firable from E.

$(M^{\uparrow}, D^{\uparrow})$, then $\widetilde{E} \overset{t_f}{\rightsquigarrow} \widetilde{E^{\uparrow}} = (M^{\uparrow}, \widetilde{D^{\uparrow}})$ and $\rceil D^{\uparrow} \lceil \subseteq \left\rceil \overrightarrow{D^{\uparrow}} \right\lceil = \left\rceil \widetilde{D^{\uparrow}} \right\lceil$.

The clause (1) holds since the system D^0 is in DBM ; we have by definition : $\rceil D^0 \lceil = \left\rceil \overrightarrow{D^0} \right\lceil = \left\rceil \widetilde{D^0} \right\lceil$. Let us prove now the clause (2). For this effect, we write: t_f the transition to fire, t_h an inhibited transition of $Ti(M)$, and t_a an activated transition of $Ta(M) - \{t_f\}$. The system D stands as $D = \overrightarrow{D} \wedge \widehat{D}$, and wherein we suppose that the system $\overrightarrow{D}$ is the full system of all DBM normalized inequalities, given as follows:

$$\left\{ \begin{array}{l} C_1 : \left\{ \begin{array}{ll} \forall t_{h_1} \neq t_{h_2} & \underline{t_{h_2}} - \underline{t_{h_1}} \leq \overrightarrow{D}[t_{h_1}, t_{h_2}] \\ \forall t_h & -\overrightarrow{D}[t_h, \bullet] \leq \underline{t_h} \leq \overrightarrow{D}[\bullet, t_h] \end{array} \right. \\ C_2 : \left\{ \begin{array}{ll} \forall t_{a_1} \neq t_{a_2} & \underline{t_{a_2}} - \underline{t_{a_1}} \leq \overrightarrow{D}[t_{a_1}, t_{a_2}] \\ \forall t_a & -\overrightarrow{D}[t_a, \bullet] \leq \underline{t_a} \leq \overrightarrow{D}[\bullet, t_a] \end{array} \right. \\ C_3 : \left\{ \begin{array}{ll} \forall t_a, t_h & \underline{t_a} - \underline{t_h} \leq \overrightarrow{D}[t_h, t_a] \\ \forall t_a, t_h & \underline{t_h} - \underline{t_a} \leq \overrightarrow{D}[t_a, t_h] \end{array} \right. \end{array} \right. \qquad \left\{ \begin{array}{l} C_4 : \left\{ \begin{array}{ll} \forall t_a & \underline{t_a} - \underline{t_f} \leq \overrightarrow{D}[t_f, t_a] \\ \forall t_a & \underline{t_f} - \underline{t_a} \leq \overrightarrow{D}[t_a, t_f] \end{array} \right. \\ C_5 : \left\{ \begin{array}{ll} \forall t_h & \underline{t_h} - \underline{t_f} \leq \overrightarrow{D}[t_f, t_h] \\ \forall t_h & \underline{t_f} - \underline{t_h} \leq \overrightarrow{D}[t_h, t_f] \end{array} \right. \\ C_6 : -\overrightarrow{D}[t_f, \bullet] \leq \underline{t_f} \leq \overrightarrow{D}[\bullet, t_f] \end{array} \right.$$

Besides, $\widetilde{D}$ is the tightest DBM over-approximation of D; hence the next property holds $(P) : \left\rceil \widetilde{D} \right\lceil = \left\rceil \overrightarrow{D} \right\lceil$.

Let us consider now the firing of the transition t_f from E to reach the class $E^{\uparrow} = (M^{\uparrow}, D^{\uparrow})$. The calculation of the system $D^{\uparrow}$ is performed by application of the algorithm given in Definition 4. We extend D to the firing constraints of t_f , $C_7 : \forall t_a \quad \underline{t_a} - \underline{t_f} \geq 0$.

Therefore, if t_f is firable from E, then the system $\rceil D \wedge C_7 \lceil \neq \emptyset$, and we have the coefficients $\overrightarrow{D}[t_f, t_a] \geq 0$ and by using the property (P), we deduce that $\widetilde{D}[t_f, t_a] \geq 0$, hence $\widetilde{\beta}[t_f] \geq 0$. Consequently, t_f is also firable from the class $\widetilde{E}$. However, it remains to prove that $\left\rceil \overrightarrow{D^{\uparrow}} \right\lceil = \left\rceil \widetilde{D^{\uparrow}} \right\lceil$.

The computation of the system $D^{\uparrow}$ is performed by replacing each variable $\underline{t_a}$ associated with an activated transition $t_a \in Ta(M) - \{t_f\}$ by $\underline{t'_a} + \underline{t_f}$. To ease the sketch of the proof, we suppose that all transitions of $Ti(M^{\uparrow}) \cup Ta(M^{\uparrow}) - \{t_f\}$ are persistent after firing t_f. Further, we limit the proof to the manipulation of DBM constraints, since we aim at computing the tightest DBM over-approximation that can be derived from D subsequently to the firing of t_f. It should be noticed that the manipulation of the constraints $\widehat{D}$ produces only new polyhedral constraints that cannot be reduced to $DBMs$ [7]. So, after substitution, the constraints of the subsystem $\overrightarrow{D} \wedge C_7$ are as follows:

$$\left\{ \begin{array}{l} C_1 \wedge C_5 \\ C'_2 : \left\{ \begin{array}{ll} \forall t_{a_1} \neq t_{a_2} & \underline{t'_{a_2}} - \underline{t'_{a_1}} \leq \overrightarrow{D}[t_{a_1}, t_{a_2}] \\ \forall t_a & -\overrightarrow{D}[t_a, \bullet] \leq \underline{t'_a} + \underline{t_f} \leq \overrightarrow{D}[\bullet, t_a] \end{array} \right. \\ C'_3 : \left\{ \begin{array}{ll} \forall t_a, t_h & \underline{t'_a} + \underline{t_f} - \underline{t_h} \leq \overrightarrow{D}[t_h, t_a] \\ \forall t_a, t_h & \underline{t_h} - \underline{t'_a} - \underline{t_f} \leq \overrightarrow{D}[t_a, t_h] \end{array} \right. \end{array} \right. \qquad \left\{ \begin{array}{l} C'_7 : -\underline{t'_a} \leq 0 \\ C'_4 : \left\{ \begin{array}{ll} \forall t_a & \underline{t'_a} \leq \overrightarrow{D}[t_f, t_a] \\ \forall t_a & -\underline{t'_a} \leq \overrightarrow{D}[t_a, t_f] \end{array} \right. \\ C_6 : -\overrightarrow{D}[t_f, \bullet] \leq \underline{t_f} \leq \overrightarrow{D}[\bullet, t_f] \end{array} \right.$$

By operating an intersection of the constraints C'_7 and C'_2, we obtain the system:

$$\begin{cases} C_1 \wedge C_2' \wedge C_3' \wedge C_4' \wedge C_5 \wedge C_6 \wedge C_7' \\ C_8 : \forall t_{a_1} \neq t_{a_2} \quad \underline{t'_{a_2}} - \underline{t'_{a_1}} \leq \overrightarrow{D}[t_{a_1}, t_{a_2}] \end{cases} \qquad C_{12} : \begin{cases} \forall t_{a_1} \quad -\underline{t'_{a_1}} \leq \underset{\forall t_{a_1} \neq t_{a_2}}{MIN} \left\{ \overrightarrow{D}[t_{a_1}, t_{a_2}] \right\} \leq 0 \\ \underline{t_f} \leq MIN_{\forall t_a} \left\{ \overrightarrow{D}[\bullet, t_a] \right\} \end{cases}$$

Then by intersection and using the property $\overrightarrow{D}[t_a, t_a] = 0$, the constraints of C_{12},C_6 C_4' change into C_6' and C_4''; we obtain:

$$\begin{cases} C_1 \wedge C_2' \wedge C_3' \wedge C_5 \wedge C_7' \wedge C_8 \\ C_6' : -\overrightarrow{D}[t_f, \bullet] \leq \underline{t_f} \leq \underset{\forall t \in Ta(M)}{MIN} \left\{ \overrightarrow{D}[\bullet, t] \right\} \\ C_4'' : \forall t_a \quad - \underset{\forall t \in Ta(M)}{MIN} \overrightarrow{D}[t_a, t] \leq \underline{t'_a} \leq \overrightarrow{D}[t_f, t_a] \end{cases}$$

We write: $\begin{cases} (F_1) : \forall t_a \quad \overrightarrow{D^{\uparrow}}[\bullet, t_a] := \overrightarrow{D}[t_f, t_a] \\ (F_2) : \overrightarrow{D^{\uparrow}}[t_a, \bullet] := \underset{\forall t \in Ta(M)}{MIN} \left\{ \overrightarrow{D}[t_a, t] \right\}. \end{cases}$

Then by using C_8 and C_4'', we obtain:

$$\begin{cases} C_1 \wedge C_2' \wedge C_3' \wedge C_4'' \wedge C_5 \wedge C_6' \wedge C_7' \\ C_8' : \forall t_{a_1} \neq t_{a_2} \quad \underline{t'_{a_2}} - \underline{t'_{a_1}} \leq MIN(\overrightarrow{D}[t_{a_1}, t_{a_2}], \overrightarrow{D^{\uparrow}}[\bullet, t_{a_2}] + \overrightarrow{D^{\uparrow}}[t_{a_1}, \bullet]) \end{cases}$$

We put

$(F_3) : \forall t_{a_1} \neq t_{a_2} \quad \overrightarrow{D^{\uparrow}}[t_{a_1}, t_{a_2}] := MIN(\overrightarrow{D}[t_{a_1}, t_{a_2}], \overrightarrow{D^{\uparrow}}[\bullet, t_{a_2}] + \overrightarrow{D^{\uparrow}}[t_{a_1}, \bullet])$.

At this stage, when the model does not contain inhibitors arcs, the system D stands as $\overrightarrow{D}$, and the constraints $C_1 \wedge C_3' \wedge C_5$ are eliminated and those of $C_2' \wedge C_6' \wedge C_7'$ can be removed as they are redundant. Therefore, the new system $D^{\uparrow}$ is given by the constraints $C_4'' \wedge C_8'$, to which we add the constraints of newly enabled transitions. The system $\widetilde{D^{\uparrow}}$ obtained from the system $\widetilde{D}$ after firing t_f can be computed in the same way as shown previously. Assuming that, if we have $D = \widetilde{D}$, then the algorithm given in Definition 6 computes an exact approximation of the system D, since the coefficients $\widetilde{D^{\uparrow}}[t_{a_1}, t_{a_2}]$ $\widetilde{D^{\uparrow}}[\bullet, t_a]$ and $\widetilde{D^{\uparrow}}[t_a, \bullet]$ are computed also by using respectively the formulae F_3, F_1 and F_2.

On the other hand, in presence of inhibited transitions, we need to operate additional manipulations on constraints C_1, C_2', C_3' and C_5 :

By intersection of the constraints of C_6' and those of C_5, of C_3' and those of C_7', and finally, of C_3' and those of C_6'; we obtain respectively the new constraints C_5', C_9 and C_{10}:

$$\begin{cases} C_1 \wedge C_2' \wedge C_3' \wedge C_4'' \wedge C_6' \wedge C_7' \wedge C_8' \\ C_5' : \begin{cases} \forall t_h \quad \underline{t_h} \leq \overrightarrow{D}[t_f, t_h] + \overrightarrow{\beta}[\bullet] \\ \forall t_h \quad -\underline{t_h} \leq \overrightarrow{D}[t_h, t_f] + \overrightarrow{D}[t_f, \bullet] \end{cases} \end{cases} \qquad \begin{array}{l} C_9 : \forall t_h \quad \underline{t_f} - \underline{t_h} \leq \overrightarrow{D}[t_h, t_a] \\ C_{10} : \begin{cases} \forall t_a, t_h \quad \underline{t'_a} - \underline{t_h} \leq \overrightarrow{D}[t_h, t_a] + \overrightarrow{D}[t_f, \bullet] \\ \forall t_a, t_h \quad \underline{t_h} - \underline{t'_a} \leq \overrightarrow{D}[t_a, t_h] + \overrightarrow{\beta}[\bullet] \end{cases} \end{array}$$

with $\forall x \in Te(M) \cup \{\bullet\}$, $\overrightarrow{\beta}[x] = \underset{\forall t \in Ta(M)}{MIN} \left\{ \overrightarrow{D}[x, t] \right\}$

Then by intersection of the constraints of C_9 with those of C_6', we obtain the constraints C_9'':

$$\begin{cases} C_1 \wedge C_2' \wedge C_3' \wedge C_4'' \wedge C_5' \wedge C_6' \wedge C_7' \wedge C_8' \wedge C_{10} \\ C_9'' : \forall t_h, \quad -\underline{t_h} \leq \overrightarrow{D}[t_h, t_a] + \overrightarrow{D}[t_f, \bullet] \end{cases}$$

By using the constraints of C_9'',C_5' and C_1 we obtain:

$$\begin{cases} C'_2 \wedge C'_3 \wedge C''_4 \wedge C'_6 \wedge C'_7 \wedge C'_8 \wedge C_{10} \\ C'_1 : \begin{cases} \forall t_{h_1} \neq t_{h_2} & \underline{t_{h_2}} - \underline{t_{h_1}} \leq D^{\uparrow}[t_{h_1}, t_{h_2}] \\ \forall t_h & -\overrightarrow{D^{\uparrow}}[t_{h_1}, \bullet]) \leq \underline{t_h} \leq \overrightarrow{D^{\uparrow}}[\bullet, t_{h_2}] \end{cases} \end{cases}$$

with

$$F_4 : \forall t_h, \quad \overrightarrow{D^{\uparrow}}[t_h, \bullet] := MIN \begin{pmatrix} \overrightarrow{D}[t_h, \bullet] \\ \overrightarrow{D}[t_f, \bullet] + \overrightarrow{\beta}[t_h] \end{pmatrix}$$

$$F_5 : \forall t_h, \quad \overrightarrow{D^{\uparrow}}[\bullet, t_h] := MIN \begin{pmatrix} \overrightarrow{D}[\bullet, t_h] \\ \overrightarrow{D}[t_f, t_h] + \overrightarrow{\beta}[\bullet] \end{pmatrix}$$

$$F_6 : \forall t_{h_1} \neq t_{h_2} \quad \overrightarrow{D^{\uparrow}}[t_{h_1}, t_{h_2}] := MIN(\overrightarrow{D}[t_{h_1}, t_{h_2}], \overrightarrow{D^{\uparrow}}[\bullet, t_{h_2}] + \overrightarrow{D^{\uparrow}}[t_{h_1}, \bullet]).$$

To achieve the proof, we proceed to the intersection of the constraints of C_{10} with those of C'_1 and C''_4; we obtain the constraints C'_{10}:

$$\begin{cases} C'_1 \wedge C'_2 \wedge C'_3 \wedge C''_4 \wedge C'_6 \wedge C'_7 \wedge C'_8 \\ C'_{10} : \begin{cases} \forall t_a, t_h & \underline{t'_a} - \underline{t_h} \leq \overrightarrow{D^{\uparrow}}[t_h, t_a] \\ \forall t_a, t_h & \underline{t_h} - \underline{t'_a} \leq \overrightarrow{D^{\uparrow}}[t_a, t_h] \end{cases} \end{cases}$$

$$(F_7) : \forall t_h \forall t_a \quad \overrightarrow{D^{\uparrow}}[t_h, t_a] := MIN(\overrightarrow{D}[t_h, t_a] + \overrightarrow{D}[t_f, \bullet], \quad \overrightarrow{D^{\uparrow}}[\bullet, t_a] + \overrightarrow{D^{\uparrow}}[t_h, \bullet]).$$

$$(F_8) : \forall t_h \forall t_a \quad \overrightarrow{D^{\uparrow}}[t_a, t_h] := MIN(\overrightarrow{D}[t_a, t_h] + \overrightarrow{\beta}[\bullet], \quad \overrightarrow{D^{\uparrow}}[\bullet, t_h] + \overrightarrow{D^{\uparrow}}[t_a, \bullet]).$$

The remaining manipulations allow to eliminate the transition t_f, thereby producing only polyhedral constraints that cannot fit into $DBMs$. These manipulations consists in the intersection of the constraints wherein the variable $\underline{t_f}$ occurs: C'_2, C'_3 and C'_6. Therefore, the system $\overrightarrow{D^{\uparrow}}$ is by construction the much precise DBM system that we can derive from $\overrightarrow{D}$ subsequently to the firing of the transition t_f. Further, assuming that the same algorithm is used to compute the coefficients of the system $\overrightarrow{D^{\uparrow}}$ as well as those of the system $\widetilde{D^{\uparrow}}$, then it is obvious that $\left\lceil \widetilde{D^{\uparrow}} \right\lceil = \left\lceil \overrightarrow{D^{\uparrow}} \right\lceil$; the property (P) holds for the systems $\widetilde{D^{\uparrow}}$ and $\overrightarrow{D^{\uparrow}}$. What is more, by assuming the formulae given previously, we prove that if $\widetilde{D}$ is in its normal form then the system $\widetilde{D^{\uparrow}}$ is also in normal form. Put in other way, as the initial class is in normal form, this guarantees, on a hand, that the DBM over-approximation is the tightest that we can compute from $\widetilde{D}$ subsequently to the firing of t_f. On the other hand, this implies also that the number of DBM that the algorithm can compute is finite, since all reachable approximated classes of the graph are in normal form [7]. □

Furthermore, the last algorithm should be provided with class equivalence conditions, in order to put an end to the enumeration process when the net is bounded. These conditions are based generally on the equality of markings and systems, as defined next:

Definition 7. *Two classes* $\widetilde{E} = (M, \widetilde{D})$ *and* $\widetilde{E'} = (M', \widetilde{D'})$, *reachable in* $\widetilde{GR}$ *satisfying the following conditions, are equivalent, and we write* $\widetilde{E} = \widetilde{E'}$:
$(i) \quad M = M' \qquad (ii) \quad \forall x, y \in (Te(M) \cup \{\bullet\})^2 \quad \widetilde{D}[x, y] = \widetilde{D'}[x, y].$

It should be noticed that the finiteness of the exact state class graph is undecidable even for bounded nets [5]. However, the graph obtained by *DBM* over-approximation is ensured to be finite when the net is bounded. This makes it possible to compute a finite *DBM* over-approximation when the exact one does not terminate. The approaches defined in [7][15] admit also that the *DBM* over-approximation that they compute, are the tightest possible. Nevertheless, these techniques have an additional cost comparatively to our algorithm. Concretely, these approaches proceed first to compute the polyhedra in its normal form (whose representation in memory and manipulation are costly), before removing the non *DBM* constraints and normalizing the *DBM*s. Then the process ends by computing the minimal form of the final *DBM* system. The normalization and the minimization induce a non neglectable computation effort. This affects the performances of the implementation of the *DBM* over-approximation.

Furthermore, the implementation in *ROMEO* [17] of the approach defined in [15] does not compute the tightest *DBM* over-approximation in much of the cases. In actual fact, *ROMEO* computes, first, the system D in its minimal form, to apply then the normalization to the set of *DBM* constraints. This makes it impossible to improve the performances of the tool but at the expenses of the precision of the *DBM* over-approximation. To highlight this point, let us consider the example of *Figure 3* already introduced in [7]. This example models three independent tasks that are conflicting for a common resource (CPU): Two periodic tasks 1 and 3 (of period 50 and 150 time units), and one sporadic task with a minimum and maximum inter-arrival times of $[100, 150]$. The task 1 (modeled by the transitions t_1 and t_4), has a higher priority than that of two other tasks, and the sporadic task has a higher priority than that of the third task. The priorities are modeled by using inhibitor arcs.

So, starting from the initial class, the firing of the sequence $(t_4, t_5, t_1, t_4, t_6, t_1, t_2, t_4, t_5, t_1, t_3, t_4, t_6, t_1, t_2, t_4, t_5, t_1, t_4, t_1, t_3, t_4, t_6, t_2, t_5, t_1, t_4, t_1, t_4, t_1, t_3, t_4, t_2)$ yields the classes $\widetilde{E}$, $\widetilde{E''}$ and E by using respectively, the algorithm introduced in this paper, the *DBM* over-approximation implemented in the tool *ROMEO* and finally the exact approach based on polyhedral representation [13] implemented also in *ROMEO*.

$\widetilde{E}=$

$M:p_2,p_3\rightarrow 1$

$\widetilde{D}$	•	t_1	t_2	t_3	t_5	t_6
•	0	40	150	140	20	28
t_1	-18	0	132	100	2	-2
t_2	-100	-60	0	40	-80	-72
t_3	-118	-100	32	0	-98	-102
t_5	-18	22	132	122	0	10
t_6	0	20	150	120	20	0

$E=\widetilde{E''}=$

$M:p_2,p_3\rightarrow 1$

$\widetilde{D''}$:

$$\begin{cases} 18 \leq \underline{t_1} \leq 40 & \underline{t_3}-\underline{t_6} \leq 120 \\ 100 \leq \underline{t_2} \leq 150 & 0 \leq \underline{t_6} \leq 28 \\ 118 \leq \underline{t_3} \leq 140 & \underline{t_6}-\underline{t_1} \leq -2 \\ 18 \leq \underline{t_5} \leq 20 & \underline{t_6}-\underline{t_3} \leq -102 \\ \underline{t_1}-\underline{t_3} \leq -100 & \\ \underline{t_1}-\underline{t_6} \leq 20 & \\ \underline{t_3}-\underline{t_1} \leq 100 & \end{cases}$$

We notice at this stage that all the resulted classes induce the same firing space whatever the approach we use. In the obtained classes, only the transition t_6 is inhibited and its maximal residual time is evaluated to 28. Now, let us consider the

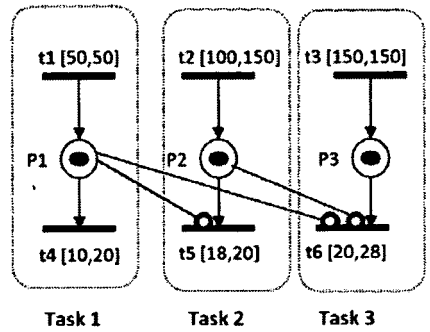

Figure 3: *ITPN modeling two periodic tasks and a sporadic one.*

firing of the transition t_1 from the previous classes to reach respectively the classes $\widetilde{E^1}$, $\widetilde{E^{"1}}$ and E^1.

$\widetilde{E^1} =$

$M^1 : p_1, p_2, p_3 \rightarrow 1$

$\widetilde{D_1}$	•	t_1	t_2	t_3	t_4	t_5	t_6
•	0	50	132	100	20	2	18
t_1	-50	0	82	50	-30	-48	-32
t_2	-80	-30	0	20	-60	-80	-62
t_3	-100	-50	32	0	-80	-98	-82
t_4	-10	40	122	90	0	-8	8
t_5	0	50	132	100	20	0	18
t_6	0	50	132	100	20	2	0

$E^1 =$

$$\begin{cases} M^1 : p_1, p_2, p_3 \rightarrow 1 \\ D^1 : \begin{cases} 50 \leq \underline{t_1} \leq 50 \\ \underline{t_2} \leq 132 \\ 100 \leq \underline{t_3} \leq 100 \\ \underline{t_5} + \underline{t_6} \leq 18 \\ \underline{t_5} - \underline{t_2} \leq -80 \\ 0 \leq \underline{t_5} \leq 2 \\ 10 \leq \underline{t_4} \leq 20 \\ 0 \leq \underline{t_6} \\ \underline{t_2} + \underline{t_6} \leq 148 \end{cases} \end{cases}$$

$$\widetilde{E^{"1}} = \begin{cases} M^1 : p_1, p_2, p_3 \rightarrow 1 \\ \widetilde{D^{"1}} : \begin{cases} 50 \leq \underline{t_1} \leq 50 & 0 \leq \underline{t_5} \leq 2 \\ 80 \leq \underline{t_2} \leq 132 & 0 \leq \underline{t_6} \leq 28 \\ 100 \leq \underline{t_3} \leq 100 & \underline{t_5} - \underline{t_2} \leq -80 \\ 10 \leq \underline{t_4} \leq 20 \end{cases} \end{cases}$$

At this stage, the persistence of the inhibited transition t_6 induces polyhedral constraints that are non redundant, as represented in the system D_1. Therefore, the *DBM* restriction induces an over-approximation of the exact class E_1. However, we notice that the implementation of our algorithm computes a tighter over-approximation. Concretely, the maximal residual time of the inhibited transition t_6 has decreased to 18. This value is exactly approximated by our algorithm, while it stands at 28 by using *ROMEO*. If we look to the system D_1 given in its minimal form, we notice that this time distance is worked out from the constraints $0 \leq \underline{t_5}$ and $\underline{t_5} + \underline{t_6} \leq 18$. This latter which is of a polyhedral form is removed, as well as the constraint $\underline{t_2} + \underline{t_6} \leq 148$, to determine the system $\widetilde{D_1^{"}}$. The normalization is then applied to the set of *DBM* constraints, whereas it should be done before

removing the polyhedral constraints. By proceeding in this way, the *DBM* over-approximation implemented in *ROMEO* improves the complexity of the approach defined in [15], but at the expenses of the precision of the over-approximation.

Let us take a look at *Figure 4* to explain why the residual time of the transition t_6 has decreased during its inhibition. This figure depicts the temporal scenario of the firing sequence (t_3, t_4, t_2, t_1) leading to the class $\widetilde{E_1}$.

From the class enabling t_1 for the first time, the transition t_3 is fired without any delay. Subsequently, the newly enabled transition t_6 is inhibited, and we have $\widetilde{D}[\bullet, t_6] = 28$. The firing of t_4 between $[10, 20]$ yields a new class. There, t_6 becomes activated for the first time and its maximal residual time $\widetilde{D}[\bullet, t_6] = 28$ stands as it was. The transition t_6 is then inhibited again after the firing of t_2 during $[0, 22]$.

Then, to be able to fire the persistent transition t_1 it needs to let time progress with minimum $tmin(t_1) = 50$ from the start of the sequence. Within this intention, only the states (valuations), that fire t_2 during $[10, 22]$ (while t_6 was activated), are satisfying the firing constraints of t_1. This restriction implies that the maximal residual time of t_6 decreases of 10 units (from 28 to 20), after the firing of t_1.

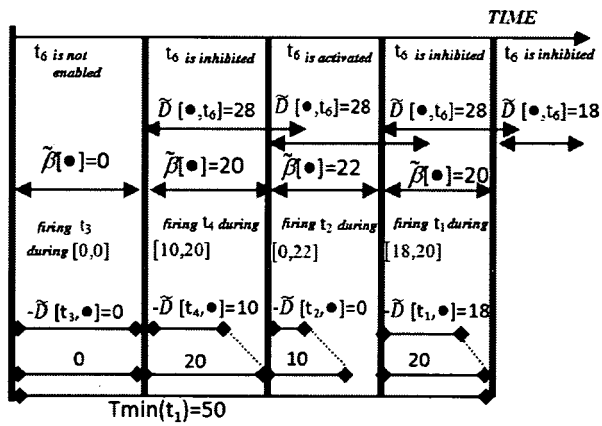

Figure 4: Temporal scenario of the firing sequence (t_3, t_4, t_2, t_1).

The loss of precision reported in the *DBM* over-approximation implemented in the tool *ROMEO* can have an impact on the firability of yet persistent transitions. To illustrate this point, let us consider, for instance, the firing of the sequence (t_4, t_5) from the classes $\widetilde{E^1}$, $\widetilde{E''^1}$ and E^1, leading respectively to the classes $\widetilde{E^2}$,$\widetilde{E''^2}$ and E^2.

$\widetilde{E^2} =$ $M^2 : p_3 \rightarrow 1$

$\widetilde{D^2}$	•	t_1	t_2	t_3	t_6
•	0	40	122	90	18
t_1	-28	0	82	50	-10
t_2	-60	-30	0	20	-42
t_3	-78	-50	32	0	-60
t_6	0	40	122	90	0

$\widetilde{E^{"2}} :=$

$M^2 : p_3 \rightarrow 1$

$\widetilde{D^{"2}}$:

$28 \leq \underline{t_1} \leq 40$ $\quad \underline{t_1} - \underline{t_3} \leq$ -50

$60 \leq \underline{t_2} \leq 122$ $\quad \underline{t_2} - \underline{t_1} \leq 82$

$78 \leq \underline{t_3} \leq 90$ $\quad \underline{t_2} - \underline{t_3} \leq 32$

$0 \leq \underline{t_6} \leq 28$ $\quad \underline{t_3} - \underline{t_1} \leq 50$

$\underline{t_1} - \underline{t_2} \leq$ -30 $\quad \underline{t_3} - \underline{t_2} \leq 20$

$E^2 :=$

$M^2 : p_3 \rightarrow 1$

D^2 :

$\underline{t_1} - \underline{t_3} \leq$ -50 $\quad \underline{t_6} \leq 18$

$28 \leq \underline{t_1} \leq 40$ $\quad 0 \leq \underline{t_6}$

$\underline{t_1} - \underline{t_2} \leq$ -30 $\quad 60 \leq \underline{t_2}$

$\underline{t_6} + \underline{t_2} - \underline{t_1} \leq 98$ $\quad \underline{t_6} - \underline{t_1} \leq$ -12

$\underline{t_2} - \underline{t_1} \leq$ -82

We notice that the transition t_1 is firable from the approximated class $\widetilde{E^{"2}}$, while it is not from the exact class E^2 (since $t_6 - \underline{t_1} \leq$ -12), as well as when using our algorithm (since $\widetilde{D^2}[t_1, t_6] = -10 < 0$). This ensures that our construction yields a less coarse over-approximation than that implemented in *ROMEO*.

Comparatively to the implementation of the approach of [7] in *ORIS* tool [16], the resulted *DBM* systems are equivalent to the ones computed by our algorithm. Actually, the implemented algorithm in *ORIS* proceeds by normalizing the pre-computed systems before removing the non *DBM* constraints. However, by proceeding in this way the durations needed to compute the graphs are higher as we will see in section 5 when reporting the simulation results. Moreover, although the graphs computed by *ORIS* are different, they are bisimilar to those computed by our algorithm. Actually, a class is expressed in *ORIS* as a tuple $(M, \widetilde{D}, New)$ where *New* is a boolean function which indicates whether each transition enabled in the class is persistent or not. As the equivalence test implemented in *ORIS* is based on the equality of all the three parameters[7], this construction therefore yields coarser graphs.

In respect to the approach defined in [5], the latter proceeds by quantization, to over-approximate the exact polyhedra by another one. This approach is configurable, namely the precision of the approximation (grid), can be fixed before the graph computation. When using the thinnest grid, this approach succeeds to compute the exact graph in almost all cases but with the highest cost in terms of computation time and memory usage. However, as this technique still manipulates general polyhedra, it undergoes a higher computation complexity comparatively to *DBM* over approximation.

To sum up, the exact computation as well as the K-grid approximation of the graph of the net of *Figure 3* yield 320 classes. On the other hand, the *DBM*

[7] Only the equality of the parameters M and $\widetilde{D}$ is needed to decide the equivalence, as evidenced in Definition 7.

over-approximation computed by *ROMEO* contains 403 classes and that by using *ORIS* produces 429 classes. The implementation of the construction defined in this paper produces a more precise and compact graph of 394 classes.

We explore in the next section how to improve the construction of the *DBM* over-approximation. This is achieved by reducing still more its size and the effort of its computation, but however while relaxing a little bit in the precision of the over-approximation.

4 Effiient construction of a DBM over-approximation

We have proposed in the last section an efficient algorithm to compute the tightest *DBM* over-approximation of an *ITPN*. We have shown that the firing interval $[-\widetilde{D}[t,\bullet], \widetilde{D}[\bullet,t]$ of a persistent inhibited transition may narrow along a firing sequence. However, in almost all cases this interval remains unchanged. Therefore, we propose in the sequel to relax a little bit the *DBM* constraints so that to compute more compact graphs while reducing their computation effort. However, as this new abstraction is not as precise as that defined in Definition 6, it may therefore contain additional sequences. On the other hand, it makes it possible to compute in many cases, with lesser expenses, more compact graphs that are indeed bisimilar to $\widetilde{GR}$ and GR. Formally, this construction is defined as follows:

Definition 8. *The contracted DBM over-approximation graph of an ITPN, denoted by* $\widetilde{GRC}$, *is the tuple* $(\widetilde{CEC}, \widetilde{E_c^0}, \hookrightarrow)$, *such that :*

- $\widetilde{CEC}$ *is the set of DBM over-approximated classes reachable in* $\widetilde{GRC}$;
- $\widetilde{E_c^0} = (M^0, \widetilde{D_c^0}) \in \widetilde{CEC}$ *is the initial class such that* $\widetilde{D_c^0} = \widetilde{D^0} = D^0$.
- $\hookrightarrow$ *is a transition relation between DBM over-approximated classes defined on* $\widetilde{CEC} \times T \times \widetilde{CEC}$, *such that* $((M, \widetilde{D_c}), t_f, (M^\uparrow, \widetilde{D_c^\uparrow})) \in \hookrightarrow$, *iff :*
  - $(t_f \in Ta(M)) \wedge (\widetilde{\beta_c}[t_f] \geq 0)$ *such that:* $\forall x \in Te(M) \cup \{\bullet\}, \widetilde{\beta_c}[x] = \underset{\forall t \in Ta(M)}{MIN} \left\{\widetilde{D_c}[x,t]\right\}$.
  - $\forall p \in P,\ M^\uparrow(p) := M(p) - B(p,t_f) + F(p,t_f)$.
  - *The coefficients of the DBM inequalities of the system* $\widetilde{D_c^\uparrow}$ *are computed from those of* $\widetilde{D_c}$ *by applying the following algorithm:*
    $\forall t \in Te(M^\uparrow)$
    $\widetilde{D_c^\uparrow}[t,t] := 0; \qquad \widetilde{D_c^\uparrow}[\bullet,\bullet] := 0.$
    If t is persistent

$$\textit{If } t \in Ti(M) \qquad \widetilde{D_c^{\uparrow}}[t,\bullet] := \widetilde{D_c}[t,\bullet]; \qquad \widetilde{D_c^{\uparrow}}[\bullet,t] := \widetilde{D_c}[\bullet,t].$$

$$\textit{If } t \notin Ti(M) \qquad \widetilde{D_c^{\uparrow}}[\bullet,t] := \widetilde{D_c}[t_f,t]\ ; \qquad \widetilde{D_c^{\uparrow}}[t,\bullet] := \widetilde{\beta_c}[t].$$

If t is newly enabled.

$$\widetilde{D_c^{\uparrow}}[\bullet,t] := tmax(t)\ ; \qquad \widetilde{D_c^{\uparrow}}[t,\bullet] := -tmin(t).$$

$$\forall (t_1,t_2) \in (Te(M^{\uparrow}))^2 \wedge (t_1 \neq t_2)$$

If t_1 or t_2 are newly enabled. $\quad \widetilde{D_c^{\uparrow}}[t_1,t_2] := \widetilde{D_c^{\uparrow}}[\bullet,t_2] + \widetilde{D_c^{\uparrow}}[t_1,\bullet].$

If t_1 and t_2 are persistent.

If $(t_1,t_2) \notin (Ti(M))^2$ or $(t_1,t_2) \in (Ti(M))^2$

$$\widetilde{D_c^{\uparrow}}[t_1,t_2] := MIN(\widetilde{D_c}[t_1,t_2],\quad \widetilde{D_c^{\uparrow}}[\bullet,t_2] + \widetilde{D_c^{\uparrow}}[t_1,\bullet]).$$

If $(t_1,t_2) \notin (Ti(M))^2 \wedge (t_1 \in Ti(M)) \vee (t_2 \in Ti(M))$

$$\widetilde{D_c^{\uparrow}}[t_1,t_2] := \widetilde{D_c^{\uparrow}}[\bullet,t_2] + \widetilde{D_c^{\uparrow}}[t_1,\bullet].$$

Compared to the construction introduced in Definition 6, the last algorithm relaxes the constraints of persistent inhibited transitions. The firing interval of the latter is assumed unchanging, while the dwelling time in the class $\widetilde{E_c}$ is neglected when computing the firing distance between a persistent inhibited transition and a persistent activated transition. However, although this construction relaxes the constraints of each class of $\widetilde{GR}$, we need to prove formally that it computes in all cases an over-approximation of $\widetilde{GR}$, and hence of GR.

Theorem 2. *The graph $\widetilde{GRC} = (\widetilde{CE}, (M^0, \widetilde{D_c^0}), \hookrightarrow)$ is a DBM over-approximation of the graph $GR = (CE, (M^0, D^0), \longmapsto)$.*

Proof. The proof is conducted in the same way as for the proof of *Theorem*.1. First of all, we notice that $\left\rceil D^0 \right\lceil = \left\rceil \overrightarrow{D^0} \right\lceil = \left\rceil \widetilde{D^0} \right\lceil = \left\rceil \widetilde{D_c^0} \right\lceil$.

Then by performing the same manipulations on the system $\overrightarrow{D}$ we determine the system:

$$\left\{ \begin{array}{l} C_1 \wedge C_2' \wedge C_3' \wedge C_4'' \wedge C_6' \wedge C_7' \wedge C_8' \\ C_5' : \left\{ \begin{array}{ll} \forall t_h & \underline{t_h} \leq \overrightarrow{D}[t_f,t_h] + \overrightarrow{\beta}[\bullet] \\ \forall t_h & -\underline{t_h} \leq \overrightarrow{D}[t_h,t_f] + \overrightarrow{D}[t_f,\bullet] \end{array} \right. \end{array} \right. \qquad C_9 : \forall t_h \quad \underline{t_f} - \underline{t_h} \leq \overrightarrow{D}[t_h,t_a]$$

$$C_{10} : \left\{ \begin{array}{ll} \forall t_a, t_h & \underline{t_a'} - \underline{t_h} \leq \overrightarrow{D}[t_h,t_a] + \overrightarrow{D}[t_f,\bullet] \\ \forall t_a, t_h & \underline{t_h} - \underline{t_a'} \leq \overrightarrow{D}[t_a,t_h] + \overrightarrow{\beta}[\bullet] \end{array} \right.$$

Then the constraints of the system $\widetilde{D_c^{\uparrow}}$ are obtained by cutting off the constraints of C_5' and C_{10}. Hence we obtain a system that is less precise than $\widetilde{D^{\uparrow}}$ and $\overrightarrow{D^{\uparrow}}$ and which defines an over-approximation of the system $\overrightarrow{D^{\uparrow}}$: $\left\rceil \overrightarrow{D^{\uparrow}} \right\lceil = \left\rceil \widetilde{D^{\uparrow}} \right\lceil \subseteq \left\rceil \widetilde{D_c^{\uparrow}} \right\lceil$. □

We show in the sequel how the construction of the graph $\widetilde{GRC}$, as defined in Definition 8, can be improved still more by reducing as well as its size as the effort of its computation. For this effect, we explore hereafter an equivalence relation

that is less restrictive than the equality which we prove to be a bisimulation. This means that the graphs resulted by using this bisimulation induces the same firing sequences as when using the equality.

We first give clues about the concepts used in this bisimulation, then we define it formally:

1. From Definition 8, we notice that the firing condition of a transition t_f depends only on the sign of the coefficients $\widetilde{D_c}[t_f, t]$. Furthermore, the computation formulae of the system $\widetilde{D_c^{\uparrow}}$ from $\widetilde{D_c}$ does not use the elements $\widetilde{D_c}[\bullet, t]$ as well as $\widetilde{D_c}[t, \bullet]$ for $t \in Ta(M)$. Therefore, as the latter coefficients are not involved in the firing tests, we need not to compare them when performing the class' equivalence test. This property makes it possible to gather classes that are not equal in the graph $\widetilde{GRC}$ but which enjoy indeed the same firing sequences.

2. Let us explore now other firing distances that are useless for the enumeration process. In order to investigate this point, first we need to introduce the following notation:

   - A transition t_i is said to be *inhibiting* t_j, if $\exists p \in P, \quad 0 < IH(p, t_i) \leq B(p, t_j)$. This means that if the transition t_j is enabled for a given marking, then t_j cannot be activated for this marking. We denote hereafter by $Inhib$ the relation defined on T^2, such that $(t_i, t_j) \in Inhib$, if t_j is *inhibiting* t_i. Note that the relation $Inhib$ is not symmetric; however if (t_i, t_j) and $(t_j, t_i) \in Inhib$, then this means that t_i and t_j are always inhibited when they are enabled together.

   Let us consider two transitions that cannot be activated for a same marking (namely, t' is inhibiting t). When building the graph $\widetilde{GRC}$, it seems obvious that the time constraints of the transition t have no impact on the firing of t', and conversely. Therefore, the distances $\widetilde{D_c}[t', t]$ and $\widetilde{D_c}[t, t']$ are useless since they are not required in the firing test since t and t' cannot be activated together. Moreover, we need not even to compute these distances and to compare them when performing the equivalence test.

3. We explore now whether some distances can be left out when dealing with conflicting transitions. Before discussing this point, we need to introduce the following notations:

   - We note AT the set of transitions of T that are not connected to any inhibitor arc: $t \in AT$, if $\nexists p \in P, IH(p, t) \neq 0$.
   - Two transitions t_i and t_j are said to be *twin*, if $\forall p \in P, \quad IH(p, t_i) = IH(p, t_j)$. This means that if two transitions are enabled for a given marking, then they are both either inhibited or activated for this marking. We denote hereafter by $Twin$ the relation defined on T^2, such that

$(t_i,t_j) \in Twin$, if t_i and t_j are *twin*. Note that $AT^2 \subseteq Twin$ and we have if $(t_i,t_j) \in Twin$, then $(t_j,t_i) \in Twin$.

In [6] the authors proposed to contract the state class graph of a TPN. They proved that the firing distances between two conflicting transitions are useless when their values stand positive. Furthermore, they show that it is not required to re-compute these distances in reachable classes as long as the conflicting transitions are not disabled ahead in the firing sequence.
In actual fact, within the context of TPN, if we have $\widetilde{D_c}[t,t'] \geq 0$, then the transition t' has no impact on the firing of t as long as both remain persistent. However, if t is fired, then t' is disabled afterwards.

For an $ITPN$, the enforcement of this property may be inconsistent when dealing with conflicting transitions that are likely to be inhibited ahead in the firing sequence. For such transitions, the problem occurs, for instance, when we have two conflicting transitions t' and t activated for two different classes $\widetilde{E_c}$ and $\widetilde{E'_c}$ such that $\widetilde{D_c}[t,t'] \neq \widetilde{D_c}[t,t'] \geq 0$. If we consider these two classes as equivalent, they might not be bisimilar indeed. To be concrete, let us assume that a sequence is fired from both $\widetilde{E_c}$ and $\widetilde{E'_c}$ during which t is inhibited. Then t becomes activated in the reachable classes $\widetilde{E_c^{\uparrow}}$ and $\widetilde{E_c^{\uparrow'}}$. At this stage, the distance $\widetilde{D_c^{\uparrow}}[t,t']$ may change to negative in $\widetilde{E_c^{\uparrow}}$ but there is no guarantee that the distance $\widetilde{D_c^{\uparrow'}}[t,t']$ may change too in $\widetilde{E_c^{\uparrow'}}$. Hence, t may not be firable from $\widetilde{E_c^{\uparrow}}$, while it might be from $\widetilde{E_c^{\uparrow'}}$. Therefore, at first glance, we should restrict the application of this property only to conflicting transitions of AT; those which are not connected to any inhibitor arc. However, we show that under some assumptions the application of this property can be also extended to inhibited transitions. Actually, to validate this contraction as a bisimulation, we need to ensure that both transitions have been inhibited during the same periods of time. To guarantee that the last condition holds, we need only to assume that the conflicting transitions t' and t are *twin* .

To illustrate this bisimulation, let us consider the $ITPN$ of *Figure* 5.*a* where we have $Inhib = \{(t_4,t_3),(t_5,t_3)\}$. According to the previous discussion, the distances $\widetilde{D_c}[t_4,t_3]$, $\widetilde{D_c}[t_3,t_4]$, $\widetilde{D_c}[t_3,t_5]$ and $\widetilde{D_c}[t_5,t_3]$ should be left out during the computation of any class of the graph as well as when performing the equivalence test. Furthermore, we have $AT = \{t_1,t_2,t_3,t_6\}$ and $Twin = AT^2 \cup \{(t_4,t_5),(t_5,t_4)\}$. However, among elements of $Twin$, only transitions t_1 and t_2, on a hand, and t_4 and t_5, on the other hand, are in conflict for the initial marking. Therefore, since the distances $\widetilde{D_c^0}[t_1,t_2]$, $\widetilde{D_c^0}[t_2,t_1]$, $\widetilde{D_c^0}[t_4,t_5]$ and $\widetilde{D_c^0}[t_5,t_4]$ are positive, we need not to re-compute their values as long as the related transitions remain persistent. Furthermore, as the firing of t_1 (resp, t_4), disables t_2 (resp, t_5), and conversely, we need not too to consider these distances for the equivalence test.

The exact construction GR, the tightest DBM over-approximation $\widetilde{GR}$ and the abstraction $\widetilde{GRC}$ produce all the same graph shown in *Figure* 5.*b*. However, the

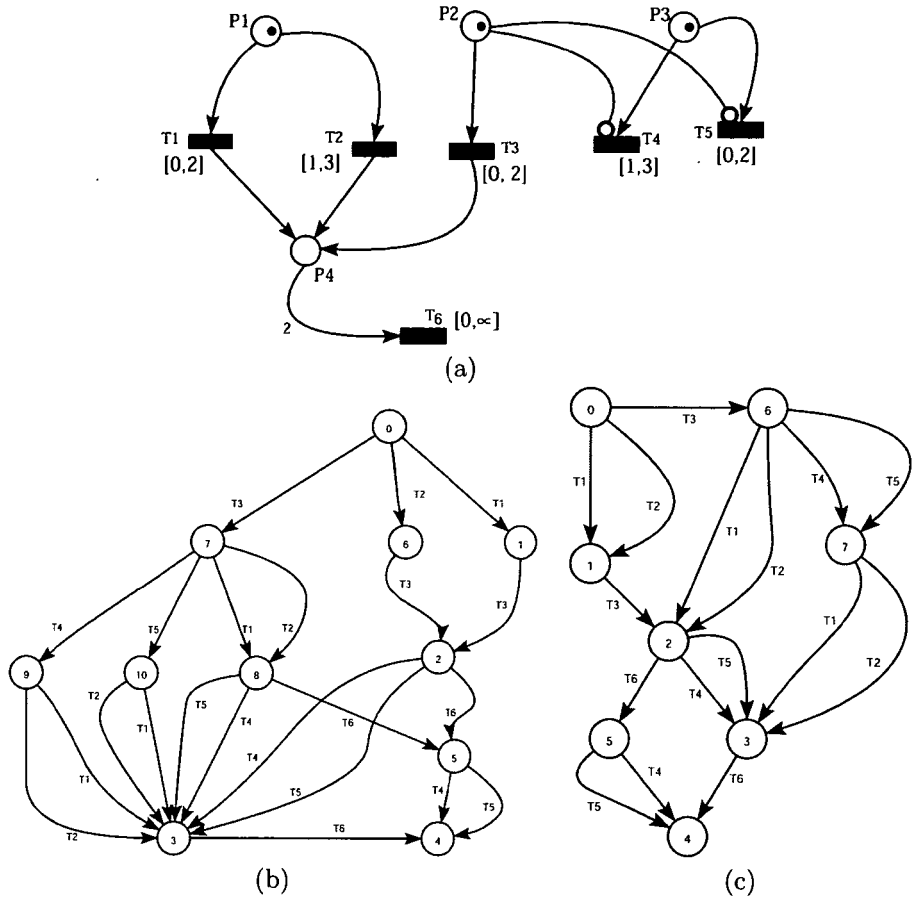

Figure 5: An *ITPN* model with twin and inhibiting transitions, with its reachability graphs.

application of the last properties makes it possible to contract further the graph $\widetilde{GRC}$, as depicted in *Figure* 5.*c*. Although it is smaller, the resulted graph still remains bisimilar to the former, as it allows gathering classes that derive from the same firing sequences[8]. For instance, firing t_1 (resp, t_2), in $\widetilde{GRC}$ from the initial class $\widetilde{E_c^0}$ leads to the class[9] $\widetilde{E_c^1}$ (resp, $\widetilde{E_c^6}$). In actual fact, the distances $\widetilde{D_c}[\bullet, t_3], \widetilde{D_c}[t_4, t_3]$ and $\widetilde{D_c}[t_5, t_3]$ which impede the equality to hold, are useless since t_3 is inhibiting t_4 and t_5. Furthermore, the classes $\widetilde{E_c^2}$ and $\widetilde{E_c^8}$ are equivalent since the value of the minimal residual time of t_4 can be left out. Finally, the classes $\widetilde{E_c^{10}}$ and $\widetilde{E_c^9}$ can be gathered since the positive distances $\widetilde{D_c}[t_1, t_2]$ can be ignored;

[8]These classes are not equal, but they are bisimilar indeed.

[9]The class $\widetilde{E_c^i}$ corresponds to the node numbered (i) in the graph.

t_1 and t_2 are two twin conflicting transitions. Hence we obtain a more compact graph of 8 nodes and 16 edges, whereas the other constructions produce a graph of 11 classes and 22 edges.

$\widetilde{E_c^0}=$ $M^0 : p_1, p_2, p_3 \rightarrow 1$

$\widetilde{D_c^0}$	•	t_1	t_2	t_3	t_4	t_5
•	0	2	3	2	3	2
t_1	0	0	3	2	3	2
t_2	-1	1	0	1	2	1
t_3	0	2	3	0	3	2
t_4	-1	2	2	1	0	1
t_5	0	2	3	2	3	0

$\widetilde{E_c^1}=$ $M^1 : p_2, p_3, p_4 \rightarrow 1$

$\widetilde{D_c^1}$	•	t_3	t_4	t_5
•	0	2	3	2
t_3	0	0	3	0
t_4	1	1	0	1
t_5	0	2	3	0

$\widetilde{E_c^6}=$ $M^6 : p_2, p_3, p_4 \rightarrow 1$

$\widetilde{D_c^6}$	•	t_3	t_4	t_5
•	0	1	3	2
t_3	0	0	3	0
t_4	1	0	0	1
t_5	0	1	3	0

$\widetilde{E_c^2}=$ $M^2 : p_4 \rightarrow 1; p_4 \rightarrow 2$

$\widetilde{D_c^2}$	•	t_4	t_5	t_6
•	0	3	2	∞
t_4	-1	0	1	∞
t_5	0	3	0	∞
t_6	0	3	2	0

$\widetilde{E_c^8}=$ $M^8 : p_4 \rightarrow 1; p_4 \rightarrow 2$

$\widetilde{D_c^8}$	•	t_4	t_5	t_6
•	0	3	2	∞
t_4	0	0	1	∞
t_5	0	3	0	∞
t_6	0	3	2	0

$\widetilde{E_c^9}=$ $M^9 : p_1, p_4 \rightarrow 1$

$\widetilde{D_c^9}$	•	t_1	t_2
•	0	1	2
t_1	0	0	2
t_2	0	1	0

$\widetilde{E_c^{10}}=$ $M^{10} : p_1, p_4 \rightarrow 1$

$\widetilde{D_c^{10}}$	•	t_1	t_2
•	0	2	3
t_1	0	0	3
t_2	0	1	0

More formally, we introduce this contraction as an equivalence relation, defined as given next:

Definition 9. *Let $\simeq$ be a relation over state classes of the graph $\widetilde{GRC}$, defined by:* $((M, \widetilde{D_c}), (M', \widetilde{D_c'})) \in \simeq$ *iff:*

(i) $M = M'$

(ii) $\forall t \in Ti(M) \quad \widetilde{D_c}[\bullet, t] = \widetilde{D_c'}[\bullet, t], \quad \widetilde{D_c}[t, \bullet] = \widetilde{D_c'}[t, \bullet]$

(iii) $\forall (t, t') \in Twin \cap Conf(M)$

$$\begin{cases} sg(\widetilde{D_c}[t, t']) = sg(\widetilde{D_c'}[t, t']) \\ \widetilde{D_c}[t, t'] = \widetilde{D_c'}[t, t'] \qquad \text{If } sg(\widetilde{D_c}[t, t']) = <_0 \end{cases}$$

(iv) $\forall (t, t') \in Te(M)^2 - (Twin \cap Conf(M))$ *such that* $(t', t), (t, t') \notin Inhib$, $\widetilde{D_c}[t, t'] = \widetilde{D_c'}[t, t']$.

where $sg(v)$ is a function which gives the sign of the value v, $sg : \mathbb{Q} \cup \{\infty\} \longrightarrow \{\geq_0, <_0\}$ such that $\geq_0$ (resp, $<_0$), denotes "positive or null" (resp, strictly negative).

In concrete terms, two classes $(M, \widetilde{D_c})$ and $(M', \widetilde{D'_c})$ are in the relation $\simeq$, iff: (i) they enjoy the same marking; (ii) the maximum and the minimum residual times of any inhibited transition must be equal in both classes; (iii) for any pair of conflicting twin enabled transitions, the firing distances involving both transitions in both classes hold the same sign, and these distances must be equal in both classes only when they are negative; (iv) For all other pairs of enabled transitions that are not in the relation $Inhib$, the firing distance involving both transitions must be equal. Let us prove now that the relation $\simeq$ is a bisimulation over the classes of the graph $\widetilde{GRC}$.

Theorem 3. *The relation $\simeq$ is a bisimulation over the graph $\widetilde{GRC}$.*

Proof. We should prove that if $(\widetilde{E_c}, \widetilde{E'_c})$ satisfies the hypotheses of Definition 9, then we have:

1 : If an activated transition t_f can fire from $\widetilde{E}_c$, then t_f can fire from $\widetilde{E'_c}$ too.

2 : If $\widetilde{E}_c \overset{t_f}{\hookrightarrow} \widetilde{E_c^{\uparrow}} \wedge \widetilde{E'_c} \overset{t_f}{\hookrightarrow} \widetilde{E_c'^{\uparrow}}$, then $(\widetilde{E_c^{\uparrow}}, \widetilde{E_c'^{\uparrow}}) \in \simeq$; $\widetilde{E_c^{\uparrow}}$ and $\widetilde{E_c'^{\uparrow}}$ satisfy Definition 9.

1. Let us assume that the transition t_f is firable from $\widetilde{E_c} = (M, \widetilde{D_c})$. As $\widetilde{E_c}$ and $\widetilde{E'_c}$ are in the relation $\simeq$, then the hypotheses of Definition 9 are satisfied. Basing on the firing condition, we need to prove that (A1): if $\widetilde{\beta_c}[t_f] \geq 0$, then $\widetilde{\beta'_c}[t_f] \geq 0$, namely that $\underset{\forall t' \in Ta(M)}{MIN} \left\{ \widetilde{D'_c}[t_f, t'] \right\} \geq 0$. As t_f and t' are both activated, then $(t_f, t'), (t', t_f) \notin Inhib$. Hence, from hypotheses (iii) and (iv) of Definition 9 we determine the property (A1); t_f is firable from $\widetilde{E'}$.

2. We have to prove that the hypotheses of Definition 9 are satisfied for $(\widetilde{E_c^{\uparrow}}, \widetilde{E_c'^{\uparrow}})$

   (a) It is obvious that as $M = M'$, we have $M^{\uparrow} = M'^{\uparrow}$.

   (b) Let us prove that $\forall t \in Ti(M^{\uparrow}), \widetilde{D_c}[\bullet, t] = \widetilde{D_c'^{\uparrow}}[\bullet, t]$. Let us replace $\widetilde{D_c^{\uparrow}}[\bullet, t]$ with its computation formula according to the status of t, as given in Definition 8.
   if $t \in New(M^{\uparrow})$, then we have :$\widetilde{D_c^{\uparrow}}[\bullet, t] = \widetilde{D_c'^{\uparrow}}[\bullet, t] = tmax(t)$.
   if $t \notin New(M^{\uparrow})$, then we should consider whether t is inhibited for M or not.

   - If $t \notin Ti(M)$, then we need to prove that $\widetilde{D_c}[t_f, t] = \widetilde{D'_c}[t_f, t]$. This last property holds since $(t_f, t) \notin Twin \cap Conf(M)$, otherwise t should be disabled after firing t_f. Furthermore, $(t_f, t) \notin Inhib$ otherwise t_f should be inhibited for M. Also $(t, t_f) \notin Inhib$, otherwise t should be inhibited for M.

- If $t \in Ti(M)$, then the proof is obvious from the hypothesis (ii).

Notice that likewise we can also prove that $\forall t \in Ta(M^{\uparrow}), \widetilde{D_c}[\bullet, t] = \widetilde{D_c'^{\uparrow}}[\bullet, t]$.

(c) We should prove that $\forall t \in Ti(M^{\uparrow}), \widetilde{D_c^{\uparrow}}[t, \bullet] = \widetilde{D_c'^{\uparrow}}[t, \bullet]$. Let us replace $\widetilde{D_c^{\uparrow}}[t, \bullet]$ with the suitable computation formula according to the status of the transition t.
if $t \in New(M^{\uparrow})$, then we have $\widetilde{D_c^{\uparrow}}[t, \bullet] = \widetilde{D_c'^{\uparrow}}[t, \bullet] = -tmin(t)$.
if $t \notin New(M^{\uparrow})$, then we should consider whether t is inhibited for M or not.

- If $t \notin Ti(M)$, then we need to prove that $\widetilde{\beta_c^{\uparrow}}[t] = \widetilde{\beta_c'^{\uparrow}}[t]$, namely that $\underset{\forall t' \in Ta(M)}{MIN} \left\{ \widetilde{D_c}[t, t'] \right\} = \underset{\forall t' \in Ta(M)}{MIN} \left\{ \widetilde{D_c'}[t, t'] \right\}$.
  If $(t, t') \notin (Twin \cap Conf(M)) \cup Inhib$, then we have $\widetilde{D_c}[t, t'] = \widetilde{D_c'}[t, t']$. However, $(t', t) \notin Inhib$ (resp, $(t, t') \notin Inhib$), otherwise t' must be inhibited (resp, t must be inhibited), for M.
  If $(t, t') \in Twin \cap Conf(M)$, then we have $sg(\widetilde{D}[t, t']) = sg(\widetilde{D'}[t, t'])$ and yet more $\widetilde{D_c}[t, t'] = \widetilde{D_c'}[t, t']$ when $sg(\widetilde{D}[t, t']) =<_0$. As $t \in Ta(M)$ and $\widetilde{D_c}[t, t] = 0$, then $\underset{\forall t' \in Ta(M)}{MIN} \left\{ \widetilde{D_c}[t, t'] \right\} \leq 0$. Therefore the value of $\widetilde{D_c}[t, t']$ has no effect on the calculation of the minimum when $sg(\widetilde{D_c}[t, t']) =\geq_0$; hence the equality holds.
- If $t \in Ti(M)$, then the proof is stemmed from the hypothesis (ii).

Notice that likewise we can also prove that $\forall t \in Ta(M^{\uparrow}), \widetilde{D_c^{\uparrow}}[t, \bullet] = \widetilde{D_c'^{\uparrow}}[t, \bullet]$.

(d) We have to prove that (A2):$\forall (t, t') \in Twin \cap Conf(M^{\uparrow})$

$$\begin{cases} sg(\widetilde{D_c^{\uparrow}}[t, t']) = sg(\widetilde{D_c'^{\uparrow}}[t, t']) \\ \widetilde{D_c^{\uparrow}}[t, t'] = \widetilde{D_c'^{\uparrow}}[t, t'] \qquad \text{If } sg(\widetilde{D_c^{\uparrow}}[t, t']) =<_0 \end{cases}$$

As we deal with safe nets, we can easily show that if two persistent transitions are in conflict for M, then they remain in conflict for $M^{\uparrow}$. Hence, $Conf(M^{\uparrow})$ consists of all the pairs of transitions $(t, t') \in Conf(M)$ that are persistent in $M^{\uparrow}$ to which we add the pairs of conflicting transitions for $M^{\uparrow}$ where at least one transition is newly enabled. First of all, it is obvious that if $t \in New(M^{\uparrow})$ or $t' \in New(M^{\uparrow})$, then (t, t') satisfies the hypothesis (A2).
Let us discuss the case where two twin conflicting transitions are persistent, $(t, t') \notin (New(M^{\uparrow}))^2$. Therefore, according to Definition 8, we have:
$\widetilde{D_c^{\uparrow}}[t, t'] = MIN(\widetilde{D_c}[t, t'], \widetilde{D_c^{\uparrow}}[t, \bullet] + \widetilde{D_c^{\uparrow}}[\bullet, t'])$.
As it is assumed that $sg(\widetilde{D_c}[t, t']) = sg(\widetilde{D_c'}[t, t'])$, and we have already

proved through $(b),(c)$ that (A3): $\forall\ t \in Te(M)$, $\widetilde{D_c^{\uparrow}}[\bullet,t] = \widetilde{D_c'^{\uparrow}}[\bullet,t]$ and $\widetilde{D_c^{\uparrow}}[t,\bullet] = \widetilde{D_c'^{\uparrow}}[t,\bullet]$, we can easily determine that $sg(\widetilde{D_c^{\uparrow}}[t,t']) = sg(\widetilde{D_c'^{\uparrow}}[t,t'])$.

Furthermore, if $sg(\widetilde{D_c^{\uparrow}}[t,t']) = sg(\widetilde{D_c'^{\uparrow}}[t,t']) =<_0$, then we should prove that $\widetilde{D_c^{\uparrow}}[t,t'] = \widetilde{D_c'^{\uparrow}}[t,t']$; two cases can be seen:

- $\widetilde{D_c^{\uparrow}}[t,t'] = \widetilde{D_c}[t,t']$: as $\widetilde{D_c}[t,t'] = \widetilde{D'}[t,t']$ when $sg(\widetilde{D_c}[t,t']) =<_0$, we guarantee that $\widetilde{D_c^{\uparrow}}[t,t'] = \widetilde{D_c'^{\uparrow}}[t,t']$.
- $\widetilde{D_c^{\uparrow}}[t,t'] = \widetilde{D_c^{\uparrow}}[t,\bullet] + \widetilde{D_c^{\uparrow}}[\bullet,t']$: the property (A3) guarantees that $\widetilde{D_c^{\uparrow}}[t,t'] = \widetilde{D_c'^{\uparrow}}[t,t']$.

(e) We have to prove that (A4): $\forall (t,t') \in Te(M^{\uparrow})^2 - ((Twin \cap Conf(M^{\uparrow}))$, such that $(t,t'),(t',t) \notin Inhib$, $\widetilde{D_c^{\uparrow}}[t,t'] = \widetilde{D_c'^{\uparrow}}[t,t']$.
If $t \in New(M^{\uparrow})$ or $t' \in New(M^{\uparrow})$, then by using property (A3) we prove that,
$\widetilde{D_c^{\uparrow}}[t,t'] = \widetilde{D_c'^{\uparrow}}[t,t'] = \widetilde{D_c^{\uparrow}}[t,\bullet] + \widetilde{D_c^{\uparrow}}[\bullet,t'] = \widetilde{D_c'^{\uparrow}}[t,\bullet] + \widetilde{D_c'^{\uparrow}}[\bullet,t']$.
Let us discuss the case where both transitions are persistent; we have either
$\widetilde{D_c^{\uparrow}}[t,t'] = MIN(\widetilde{D_c}[t,t'], \widetilde{D_c^{\uparrow}}[t,\bullet] + \widetilde{D_c^{\uparrow}}[\bullet,t'])$ or $\widetilde{D_c^{\uparrow}}[t,t'] = \widetilde{D_c^{\uparrow}}[t,\bullet] + \widetilde{D_c^{\uparrow}}[\bullet,t']$.
As $(t,t') \notin (Twin \cap Conf(M^{\uparrow}) \cup Inhib$, then $\widetilde{D_c}[t,t'] = \widetilde{D_c'}[t,t']$; hence the property (A4) holds.

□

4.1 Discussion

By avoiding, on one hand, to compute some distances when working out each reachable class, and on the other hand, to compare them during the equivalence test, we succeed in reducing the computation effort of the approximated graph $\widetilde{GRC}$. This construction achieves, in general, to significantly reduce the size of the graphs, but however loses a bit in the precision of the approximation. In fact, the relaxation of some DBM constraints in the abstraction $\widetilde{GRC}$ allows to gather many classes as equivalent even when they are not equal. On the other hand, these classes are not considered as equivalent in the other constructions GR and $\widetilde{GR}$ although they often stand bisimilar.

For example, let us go back to the net of *Figure*.3 and let us consider again the sequence leading to the classes $\widetilde{E^2}$,$\widetilde{E''^2}$ and E^2 (see page 20). Through the same sequence we reach the class $\widetilde{E_c^2}$ in the graph $\widetilde{GRC}$. As it is the case for the DBM over-approximation defined in [15], the class $\widetilde{E_c^2}$ allows the firing of transition t_1. On the other hand, the exact construction [13], the K-grid approximation [5], the

algorithm introduced in Definition.6 as well as the approach defined in [7] do not. Actually, the firing of the transition t_1 is due to the relaxation of the constraints of the transition t_6 in $\widetilde{E_c^2}$. However, despite that the abstraction $\widetilde{GRC}$ does not build the tightest *DBM* over-approximation, it succeeds in some cases to compute much compact graphs than other approaches. For instance, the resulted graph obtained for the net of *Figure*.3 contains only 309 classes.

$\widetilde{E_c^2} =$ $M^2 : p_3 \rightarrow 1$

$\widetilde{D_c^2}$	•	t_1	t_2	t_3	t_6
•	0	40	122	90	28
t_1	-28	0	82	50	0
t_2	-58	-30	0	20	-30
t_3	-78	-50	32	0	-50
t_6	0	40	122	90	0

To illustrate this fact, let us consider the firing of the transition t_6 from the previous classes $\widetilde{E^2}$,$\widetilde{E_c^2}$, $\widetilde{E"^2}$ and E^2 to reach respectively the classes $\widetilde{E^3}$,$\widetilde{E_c^3}$, $\widetilde{E"^3}$ and E^3.

$\widetilde{E^3} =$ $M^3 :\rightarrow 0$

$\widetilde{D^3}$	•	t_1	t_2	t_3
•	0	40	122	90
t_1	-10	0	82	50
t_2	-42	-30	0	20
t_3	-60	-50	32	0

$\widetilde{E_c^3} =$ $M^3 :\rightarrow 0$

$\widetilde{D_c^3}$	•	t_1	t_2	t_3
•	0	40	122	90
t_1	-28	0	82	50
t_2	-30	-30	0	20
t_3	-50	-50	32	0

$E^3 :=$

$M^3 :\rightarrow 0$

$D^3 :$

$\begin{cases} 12 \leq \underline{t_1} \leq 40 & \underline{t_1} - \underline{t_3} \leq \text{-}50 \\ \underline{t_1} - \underline{t_2} \leq \text{-}30 & \underline{t_2} - \underline{t_1} \leq 82 \end{cases}$

$\widetilde{E"^3}:=$

$M^3 :\rightarrow 0$

$\widetilde{D"^3} :$

$\begin{cases} 0 \leq \underline{t_1} \leq 40 & \underline{t_1} - \underline{t_2} \leq \text{-}30 \\ 32 \leq \underline{t_2} \leq 122 & \underline{t_1} - \underline{t_3} = \text{-}50 \\ 50 \leq \underline{t_3} \leq 90 & \underline{t_3} - \underline{t_2} \leq 20 \end{cases}$

As we can see, all the resulted classes contain only *DBM* constraints. However the ones computed by the exact approach and the K-grid based approximation are the tighter ones (see the class E^3). On the other side, the *DBM* over-approximations introduced in Definition.6 and in [7] produce the same class $\widetilde{E^3}$, whereas the one defined in [15] as well as the construction $\widetilde{GRC}$ compute less precise *DBM* systems (see $\widetilde{E"^3}$ and $\widetilde{E_c^3}$). Although all these classes are unequal, they still derive the same firing sequences. Furthermore, the classes $\widetilde{E^3}$,$\widetilde{E"^3}$ and E^3 stand unique and are not equal to any other reachable class in their related graphs. However, the use of the equivalence $\simeq$ rather than the equality in the construction of $\widetilde{GRC}$ makes it possible to gather in a same node four other classes that stand bisimilar to $\widetilde{E_c^3}$ in the graph $\widetilde{GRC}$. Actually, according to Definition 9, the constraints $0 \leq \underline{t_1} \leq 40$, $32 \leq \underline{t_2} \leq 122$ and $50 \leq \underline{t_3} \leq 90$ in the class $\widetilde{E_c^3}$ are not needed

to carry out the equivalence test. Therefore, despite the loss of precision in the over-approximation (which may induce additional sequences in the graph $\widetilde{GRC}$), this construction still computes a much compact graph than all other approaches.

Consequently, the abstraction over the classes of the graph $\widetilde{GRC}$ is the quotient graph of $\widetilde{GRC}$ w.r.t the relation $\simeq$. It preserves, markings and both firing sequences while it is, in general, smaller. The $\widetilde{GRC}$ may be more appropriate than $\widetilde{GR}$ to check over linear properties of the model, especially when the number of additional sequences that have been added due to constraint relaxation is limited. However, when the graph $\widetilde{GRC}$ provides a too coarse over-approximation, it may yield a larger graph than $\widetilde{GR}$; the additional sequences are too numerous to be wrapped by the contraction. Indeed, the construction of $\widetilde{GRC}$ is more convenient to build when many inhibiting and conflicting transitions are reported in the net, otherwise the construction of $\widetilde{GR}$ should be considered. In other respects, it should be noticed that all the sequences firable in GR are preserved in $\widetilde{GR}$ and hence in the $\widetilde{GRC}$.

5 Experimental results

We have implemented the algorithm using $C++$ builder language on a Windows workstation. The graph construction is based on breadth-first graph generation search strategy. The experiments have been performed on a Pentium V with a processor speed of $2,7$ GHZ and $1,9$ GB of memory capacity. The different tests have been carried out by using different tools: *TINA* tool[18], *ROMEO* tool[17], *ORIS* tool[16] and our tool named $ITPNT$.

The performances of the experiments are assessed by considering three parameters, the number of classes, the number of edges, and finally in terms of computation times. It is noteworthy that *ROMEO* and *ORIS* tool do not bring out some parameters; we denote that by the notation *NA* (*Not Available*). Also, we denote by *NF* (*Not Finished*) the tests that had led to memory overflows or to a big time computation; more than 5 minutes.

Through the first experiments we have checked whether the *TPN* graph con-

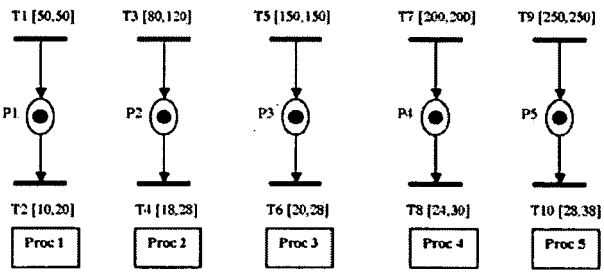

Figure 6: TPN used in the experiments.

Table 2: Results of experiments performed on TPN.

Examples	*Tools*	*TINA*	*ROMEO*	*ORIS*	*ITPNT*
Proc 1	*Classes*	2	2	2	2
	Edges	2	2	*NA*	2
	Times (ms)	0	*NA*	0	0
Proc 1 2	*Classes*	186	186	188	186
	Edges	262	262	*NA*	262
	Times (ms)	0	*NA*	16	0
Proc 1 2 3	*Classes*	958	958	1038	958
	Edges	1506	1506	*NA*	1506
	Times (ms)	1	*NA*	391	3
Proc 1 2 3 4	*Classes*	5.219	5.219	6.029	5.219
	Edges	8.580	8.580	*NA*	8.580
	Times (ms)	31	*NA*	2.719	38
Proc 1 2 3 4 5	*Classes*	42.909	42.909	52.452	42.909
	Edges	73.842	73.842	*NA*	73.842
	Times (ms)	734	*NA*	30000	786

struction by using our algorithm is conforming with that of other tools. For this effect, we have considered the combination of the TPN shown in *Figure 6*. First, we started by testing the net *Proc1,* then by combining it with *Proc2,* and so on. The results of these experiments are reported in *Table 2*. The latter shows that when assuming the equality as equivalence relation, the computed graphs are identical whatever the tool we use. However, as the expression of a class is extended to the parameter NEW in $ORIS$, the graphs computed by using this tool are coarser.

In the second series of tests, we aim at comparing the graph constructions defined in this paper with other fellow approaches. First of all, we have considered the $ITPN$ given in *Figure 3* while varying the intervals of transitions t_2,t_3 and t_6. The results of the tests are given in *Table 3*. We notice that for all the tests performed, our algorithms outperform the other tools in terms of computation time.

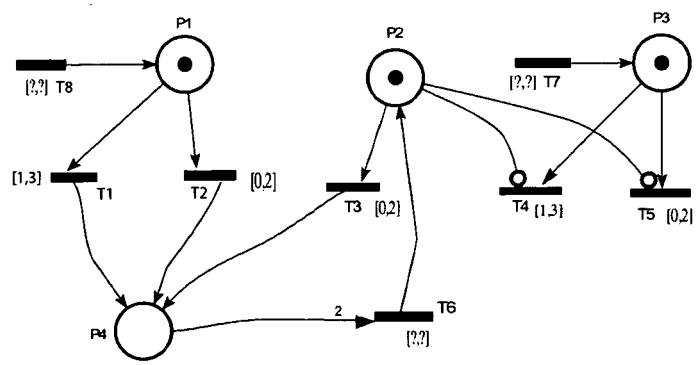

Figure 7: $ITPN$ used in the experiments.

Indeed, the optimization in the computation of DBM systems makes it possible to speed up the construction of ours graph, faster than in the other tools. Comparing to $ORIS$ (which is assumed to compute the tightest DBM over-approximation as in $ITPNT$), the computation times are reduced approximatively by 30. Concerning the tool $ROMEO$, besides that the computed graphs are less precise than ours, it seems also that the times needed for that are slower, even though they are not revealed. Moreover, the exact construction of the graph GR needed high computation times to achieve (more than 1 minute in average), even failed in many cases as evidenced in the fourth test of *Table*.3. However, in some other cases (see the two final tests in *Table*.3), the exact construction succeeds to compute finite graphs whereas all DBM-over approximations techniques fail. This happens when the additional sequences (due to over-approximation), compute persistently new markings that stand unbounded in the graph.

Table 3: Results of experiments performed with $ITPN$ of Figure 3

	TOOLS	TINA	ROMEO		ITPNT		ORIS
Examples	Methods	K-grid	Exact	DBM	$\widetilde{GR}(=)$	$\widetilde{GRC}(\simeq)$	DBM
t_2 [100,150]	Classes	4.489	4.489	5.431	5.378	5.098	5.538
t_3 [160,160]	Edges	6.360	6.360	7.608	7.530	7.251	NA
t_6 [20,28]	Times(ms)	1632	NA	NA	51	45	1578
t_2 [100,150]	Classes	320	320	403	394	309	429
t_3 [150,150]	Edges	460	460	575	562	446	NA
t_6 [20,28]	Times(ms)	110	NA	NA	2	1	156
t_2 [100,150]	Classes	4.142	4.142	5.034	4.982	4.759	5140
t_3 [140,140]	Edges	5.889	5.889	7.095	7.014	6.781	NA
t_6 [20,28]	Times(ms)	1502	NA	NA	41	37	1765
t_3 [80,120]	Classes	28 392	NF	47.622	40.842	43.462	NF
t_5 [155,155]	edges	41.452	NF	67.309	57.766	60.951	NF
t_6 [20,28]	Times (ms)	15703	NA	NA	878	886	NF
t_2 [80,120]	Classes	7.018	7.018	12.379	10.004	10.888	10.400
t_3 [140,140]	Edges	10.242	10.242	17.829	14.406	15.490	NA
t_6 [20,28]	Times(ms)	2834	NA	NA	178	162	3.516
t_2 [100,150]	Classes	11.351	11.351	16.354	15.178	16.646	15.318
t_3 [135,135]	Edges	15.649	15.649	22.230	20.486	23.225	NA
t_6 [20,28]	Times(ms)	4907	NA	NA	220	236	4765
t_2 [100,150]	Classes	17.612	17.612	21.857	21.626	22.290	21.942
t_3 [155,155]	Edges	24.522	24.522	30.065	29.711	31.151	NA
t_6 [20,28]	Times(ms)	7951	NA	NA	285	289	5594
t_2 [100,150]	Classes	12.874	12.874	NF	NF	NF	NF
t_3 [135,135]	Edges	18.424	18.424	NF	NF	NF	NF
t_6 [20,38]	Times(ms)	2340	NA	NF	NF	NF	NF
t_2 [100,150]	Classes	19.827	19.827	NF	NF	NF	NF
t_3 [155,155]	Edges	28.532	28.532	NF	NF	NF	NF
t_6 [20,39]	Times(ms)	3635	NA	NF	NF	NF	NF

Comparing to *TINA* which implements the K-grid based approximation, there are points in our favor and other against. As the results reported with *TINA* are obtained with the highest level grid, this construction achieves to build the exact graphs in almost all cases sensibly faster than *ROMEO*. However, although ours graphs are more coarse, they needed less time to be built (approximatively 25 times less). In other respects, the construction of the abstraction $\widetilde{GRC}$ seems to be more appropriate than that of $\widetilde{GR}$ when dealing with smaller graphs (less than 10.000 nodes). Otherwise the additional sequences due to the precision loss in $\widetilde{GRC}$ overwhelms the benefits of the contraction. However, the contraction is supposed to be more important in presence of conflicting transitions as we will see in the next experiments.

In the last experiments, we intend to advocate the benefits of building $\widetilde{GRC}$ rather than $\widetilde{GR}$ when dealing with both conflicting and inhibiting transitions. For this effect, we have considered the *ITPN* given in *Figure 7* while varying the intervals of transitions t_6,t_7 and t_8; the results of these experiments are reported in *Table 4*.

All the experiments show that the graph computation times are in favor of our constructions, yet more when computing the abstraction $\widetilde{GRC}$. Furthermore, in the first two experiments, the construction of the graph $\widetilde{GR}$ succeeds to build the exact graph, unlike the *DBM* over-approximations implemented in *ROMEO*.

On the other side, the construction of the abstraction $\widetilde{GRC}$ achieves to reduce significantly the size of the graphs as well as their computation effort. Moreover, this contraction has provided very compact graphs which are even smaller than those computed by the exact approach. However, this does not mean that the graphs $\widetilde{GRC}$ are more precise in the approximation than $\widetilde{GR}$, but denote that many classes that stand unequal in $\widetilde{GR}$ and GR are bisimilar indeed. The application of the equivalency $\simeq$ makes it possible to gather these bisimilar classes, and therefore to compact sensibly the graphs.

Furthermore, we notice that for the last four tests in *Table*.4, the exact computation of the graph as well as the K-grid approximation fail to build the graphs, unlike *DBM* over-approximation approaches. This happens when the number of different polyhedra computed in the exact graph is unbounded, while the number of *DBM* systems obtained by approximation is always bounded.

6 Conclusion

We have proposed in this paper an efficient algorithm to construct the tighter *DBM* over-approximation of the state class graph of preemptive systems modeled by using the *ITPN* model. Similarly as in [7][15], our approach is based on over-approximating the polyhedron of each reachable class by relaxing its non *DBM* constraints. For this effect, we have proposed to shun the computation of the intermediary polyhedra, and have provided an algorithm that computes efficiently and straightforwardly the full *DBM* system in its normal form. We have thereby

Table 4: Results of experiments performed with *ITPN* of Figure 7.

	TOOLS	TINA	ROMEO		ITPNT	
Examples	Methods	K-grid	Exact	DBM	$\widetilde{GR}(=)$	$\widetilde{GRC}(\simeq)$
t_6 [2,5]	Classes	1.035	1.035	1.318	1.035	842
t_7 [22,35]	Edges	1.830	1.830	2.323	1.830	1.471
t_8 [20,30]	Times(ms)	312	NA	NA	8	4
t_6 [2,5]	Classes	750	750	1.685	750	742
t_7 [12,15]	Edges	1.363	1.363	3.106	1.363	1.356
t_8 [10,20]	Times(ms)	219	NA	NA	4	3
t_6 [4,8]	Classes	2.346	2.346	3.398	2.402	1.880
t_7 [14,20]	Edges	4.969	4.969	7.198	5.090	4.118
t_8 [10,20]	Times(ms)	1312	NA	NA	19	12
t_6 [4,10]	Classes	3.203	3.203	4.451	3.238	2.648
t_7 [16,22]	Edges	6.603	6.603	9.184	6.756	5.758
t_8 [10,18]	Times(ms)	1.594	NA	NA	28	19
t_6 [4,8]	Classes	NF	NF	20.638	19.739	16.981
t_7 [16,22]	Edges	NF	NF	46.216	43.378	38.338
t_8 [10,15]	Times(ms)	NA	NF	NA	426	363
t_6 [4,10]	Classes	NF	NF	20.875	20.048	17.272
t_7 [16,25]	Edges	NF	NF	46.945	44.261	39.137
t_8 [10,18]	Times(ms)	NA	NF	NA	440	361
t_6 [4,10]	Classes	NF	NF	20.451	19.636	16.891
t_7 [16,25]	Edges	NF	NF	45.993	43.338	38.237
t_8 [10,20]	Times(ms)	NA	NF	NA	425	342
t_6 [4,10]	Classes	NF	NF	19.092	18.481	15.915
t_7 [16,27]	Edges	NF	NF	43.116	40.941	36.085
t_8 [10,28]	Times(ms)	NA	NF	NA	403	324

succeeded to remove the drawbacks due to the manipulation of the intermediary polyhedra, and improved significantly the graph construction by removing the cost of the normalization and the minimization of the DBM system.

Then, in the second part of this work, we have proposed a new approach to compute an abstraction of the state space of an $ITPN$. For this effect, we showed that by relaxing a little bit in the precision of the DBM over-approximation, we can compute graphs that can be more appropriate, in certain cases, to model-check the linear properties of the $ITPN$. We have discussed how this construction can be improved yet more by leaving out all the distances that are useless for the class computation process. Hence, we have put forward an equivalence relation that makes it possible to contract sensibly the size of the graphs as well as to reduce the effort of their computation. Experimental results have been reported to advocate the benefits of both constructions.

References

[1] Avis, D., K. Fukuda and S. Picozzi, On canonical representations of convex polyhedra. First International Congress of Mathematical Software (2002), pp. 350–360.

[2] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, S. Yovine: The Algorithmic Analysis of Hybrid Systems. Theor. Comput. Sci. 138(1): 3-34 (1995)

[3] Bernard Berthomieu, Miguel Menasche: An Enumerative Approach for Analyzing Time Petri Nets. IFIP Congress 1983: 41-46

[4] B. Berthomieu, and M. Diaz. "Modeling and verification of time dependant systems using Time Petri Nets". IEEE TSE, 17(3):(259-273), March 1991.

[5] Bernard Berthomieu, Didier Lime, Olivier H. Roux, François Vernadat: Reachability Problems and Abstract State Spaces for Time Petri Nets with Stopwatches. Discrete Event Dynamic Systems 17(2): 133-158 (2007).

[6] Hanifa Boucheneb, Hind Rakkay: A More Efficient Time Petri Net State Space Abstraction Useful to Model Checking Timed Linear Properties. Fundam. Inform. 88(4): 469-495 (2008).

[7] G. Bucci, A. Fedeli, L. Sassoli, and E.Vicario. Timed State Space Analysis of Real-Time Preemptive Systems. IEEE TSE, Vol 30, No. 2, Feb 2004.

[8] Dill, D.L.: Timing assumptions and verification of finite-state concurrent systems; Workshop Automatic Verification Methods for Finite-State Systems. Vol 407. (1989) 197-212.

[9] P. Merlin. "A study of the recoverability of computer system". PhD thesis Dep. Comp. Science, Uni. California, Irvine, 1974.

[10] F. Cassez and K.G. Larsen. The Impressive Power of Stopwatches. LNCS, vol. 1877, pp. 138-152, Aug. 2000.

[11] Thomas A. Henzinger: The Theory of Hybrid Automata. LICS 1996: 278-292

[12] Thomas A. Henzinger, Rupak Majumdar, Jean-François Raskin: A classification of symbolic transition systems. ACM Trans. Comput. Log. 6(1): 1-32 (2005)

[13] D.Lime, and O.H.Roux. Expressiveness and analysis of scheduling extended time Petri nets. In 5th IFAC International Conference on Fieldbus Systems and their Applications, (FET'03), Elsevier Science, July, 2003.

[14] Morgan Magnin, Didier Lime, Olivier H. Roux: An Efficient Method for Computing Exact State Space of Petri Nets With Stopwatches. Electr. Notes Theor. Comput. Sci. 144(3): 59-77 (2006).

[15] Olivier H. Roux, Didier Lime: Time Petri Nets with Inhibitor Hyperarcs. Formal Semantics and State Space Computation. ICATPN 2004: 371-390.

[16] ORIS TOOL:http://www.stlab.dsi.unifi.it/oris/index.html.

[17] ROMEO TOOL http://romeo.rts-software.org.

[18] TINA Tool http://www.laas.fr/tina/.

Received 13th January 2010

Acta Cybernetica **20** (2012) 385–398.

Density of Tautologies in Logics with One Variable

László Aszalós* and Tamás Herendi*

Abstract

In the present paper we estimate the ratio of the number of tautologies and the number of formulae of length n by determining the asymptotic density of tautologies in different kinds of logics with one variable. The logics under consideration are the ones with a single connective (*nand* or *nor*); *negation* with a connective (*disjunction* or *conjunction*); and several connectives.

Keywords: tautologies, density, generating functions

1 Introduction

One of the authors developed a computer aided education software of introductory logic, and he needed to generate formulae randomly. Naturally the following question has arisen: What percent of the randomly generated formulae are tautologies (logical law) or antilogies (contradiction)? We intend to answer this question for several logics in the present paper.

Traditionally this answer is a number expressing the ratio of favourable cases (tautologies) to the whole number of the cases possible (formulae). Since there are infinitely many tautologies, the ratio became ∞/∞, which is not suitable for us. To find a reasonable answer one have to use some approximation. At first we made some experiments, we wrote programs to list the first ten million formulae (ordered by their length) and to count the tautologies among them. The test showed that the quotient of the numbers of tautologies and formulae may have a limit. Next we calculated the numbers of different kinds of formulae up to the first 10^{100} formulae. We obtained similar results. Later we realized that Marek Zaionc gave the exact value of the asymptotic limit for some logics [10]. He, together with his collegues and students, examined purely implicational [6], intuitionistic [7, 5] and modal [3, 4] logics with a few propositional variables. Moreover he dealt with the case of several propositional variables [11]. Harris studied the k-cnf case [2]. In the present paper we provide limits for logics with one propositional variable. Our main tool for the observation of asymptotics of sequences is generating function [9]. The exposition of the method are nicely presented in the articles cited before, e.g. in [10]. Although we used the Zaionc's method, our set of logical connectives are not discussed yet.

*Faculty of Informatics, University of Debrecen, Egyetem tér 1, H4032 Debrecen, Hungary, E-mail: {Laszlo.Aszalos,herendi}@inf.unideb.hu

2 Definitions and preliminary results

In this section we introduce the concept of generating functions, and we give some basic properties of them.

Definition 1. *A complex function f is said to be analytic on a region R if it is differentiable at every point in R. It is also called regular, if it is analytic and single valued in R.*

Definition 2. *Suppose that the complex function f is analytic on a region R. Then a point z on the boundary is called regular if the extension of f on an open region containing $R \cup \{z\}$ is analytic. The point z is called singular, if it is not regular.*

Remark 1. The complex function $f(z) = \sqrt{z}$ has the only singularity at $z_0 = 0$.

Remark 2. Let $p(z)$ be a polynomial of degree d with not necessarily different roots $z_1, \ldots, z_d$. Since p is analytic on the whole complex plane, the function $\sqrt{p(z)}$ may have singularity only at the points, where $p(z) = 0$, i.e. at $z_1, \ldots, z_d$.

Definition 3. *Let u_n be a sequence of real numbers. The formal power series $U(x) = \sum_{n=0}^{\infty} u_n x^n$ is called the generating function of the sequence u_n.*

Remark 3. Let u_n, v_n and w_n be sequences of numbers with generating functions $U(x)$, $V(x)$ and $W(x)$, respectively, and let α be a real number. Then the following properties hold:

1. If $v_n = u_{n+1}$ for $n \geq 0$ then $U(x) = xV(x) + u_0$.
2. If $w_n = \sum_{i=0}^{n} u_i v_{n-i}$ for $n \geq 0$ then $W(x) = U(x)V(x)$.
3. If $w_n = u_n + v_n$ for $n \geq 0$ then $W(x) = U(x) + V(x)$.
4. If $v_n = \alpha \cdot u_n$ for $n \geq 0$ then $V(x) = \alpha \cdot U(x)$.

Theorem 1. *Let u_n and v_n be two sequences of real numbers with generating functions $U(x)$ and $V(x)$, respectively. Suppose that both $U(x)$ and $V(x)$ have singularity in x_0 and they have no other singularities in the circle $|x| \leq |x_0|$. If*

$$U(x_0 \cdot x) = \sum_{n=0}^{\infty} \hat{u}_n (1-x)^{\frac{n}{2}} \quad \text{and} \quad V(x_0 \cdot x) = \sum_{n=0}^{\infty} \hat{v}_n (1-x)^{\frac{n}{2}}$$

and $\hat{v}_1 \neq 0$ then the limit u_n/v_n exists and $\lim_{n\to\infty} u_n/v_n = \hat{u}_1/\hat{v}_1$.

Proof. In the proof we follow the arguments of Theorem 21 of [10].

By the assumed properties of $U(x)$ and $V(x)$ we may define $\bar{U}(x) = U(x_0 \cdot x)$ and $\bar{V}(x) = V(x_0 \cdot x)$ such that $\bar{U}(x)$ and $\bar{V}(x)$ are regular for $|x| < 1$ and have the only singularity in $x = 1$ on the unit circle. Hence $\bar{U}(x) = \sum_{n=0}^{\infty} \hat{u}_n (1-x)^{\frac{n}{2}}$ and $\bar{V}(x) = \sum_{n=0}^{\infty} \hat{v}_n (1-x)^{\frac{n}{2}}$. Let $\bar{u}_n = u_n/x_0^n$ and $\bar{v}_n = v_n/x_0^n$ be two sequences. Then $\bar{U}(x)$ and $\bar{V}(x)$ are the generating functions of $\bar{u}_n$ and $\bar{v}_n$, respectively. Clearly

$$\lim_{n\to\infty} \frac{\bar{u}_n}{\bar{v}_n} = \lim_{n\to\infty} \frac{u_n/x_0^n}{v_n/x_0^n} = \lim_{n\to\infty} \frac{u_n}{v_n} . \tag{1}$$

By Theorem 8.4 of [8] the asymptotic expansion of the coefficient of x^n in $\bar{U}(x)$ and $\bar{V}(x)$ are

$$\hat{u}_1\binom{1/2}{n}(-1)^n + O(n^{-2}) \quad \text{and} \quad \hat{v}_1\binom{1/2}{n}(-1)^n + O(n^{-2}),$$

respectively. Hence

$$\lim_{n\to\infty}\frac{\bar{u}_n}{\bar{v}_n} = \lim_{n\to\infty}\frac{\hat{u}_1\binom{1/2}{n}(-1)^n + O(n^{-2})}{\hat{v}_1\binom{1/2}{n}(-1)^n + O(n^{-2})}$$

Since $\lim_{n\to\infty}\binom{1/2}{n}(-1)^n = -\infty$, thus

$$\lim_{n\to\infty}\frac{\hat{u}_1\binom{1/2}{n}(-1)^n + O(n^{-2})}{\hat{v}_1\binom{1/2}{n}(-1)^n + O(n^{-2})} = \lim_{n\to\infty}\frac{\hat{u}_1\binom{1/2}{n}(-1)^n}{\hat{v}_1\binom{1/2}{n}(-1)^n} = \lim_{n\to\infty}\frac{\hat{u}_1}{\hat{v}_1} = \frac{\hat{u}_1}{\hat{v}_1}\,.$$

Combining this result with (1), we obtain the statement of the theorem. □

In the present paper we use the following method for the computation of the relative frequency of the tautologies:

1. We determine the generating function T of the number of tautologies and the generating function S of the number of formulae.
2. We calculate the minimal singularities of T and S, and we show that they are equal. We denote this common singularity by x_0.
3. We compute the expansion of $T(x_0 \cdot x)$ and $S(x_0 \cdot x)$ around the common singularity.
4. The limit of the relative frequency of the tautologies is $\hat{t}_1/\hat{s}_1$.

3 Logical foundations

Definition 4. *Let Γ be a nonempty set of unary (negation: $\neg$) and binary operators (disjunction: $\vee$, conjunction: $\wedge$, nand: $\uparrow$, nor: $\downarrow$, implication: $\supset$ and equivalence: $\equiv$) and let p be a unique propositional letter. The set of formulae of the logic of Γ is the smallest set $\mathcal{F}^\Gamma$ for which*

- $p \in \mathcal{F}^\Gamma$
- *If $\neg \in \Gamma$ and $\varphi \in \mathcal{F}^\Gamma$, then $\neg\varphi \in \mathcal{F}^\Gamma$.*
- *If $\triangle \in \Gamma$ and $\varphi, \psi \in \mathcal{F}^\Gamma$, then $(\varphi\triangle\psi) \in \mathcal{F}^\Gamma$.*

The length of a formula is defined as follows: $l(p) = 1$, $l(\neg\varphi) = l(\varphi) + 1$ and $l(\varphi\triangle\psi)) = l(\varphi) + l(\psi) + 1$.

The length of a formula is equal to the numbers of its characters excluding the parentheses (or exactly the number of its characters in polish notation).

We say that formulae φ and ψ are equivalent, if the formula $(\neg\varphi \wedge \neg\psi) \vee (\varphi \wedge \psi)$ is tautology. We have only one propositional letter p, which may have two different truth-values. Since the value of a formula can be true or false in both cases, we have four different kinds of formulae. These equivalence classes based on the equivalence defined before are denoted by their representatives: p, $\neg p$, $\top$ and $\bot$. These sets are the classes of those formulae which are equivalent with p, $\neg p$, tautologies and antilogies, respectively. We will call the members of the equivalence class of p p-formulae, and the members of the equivalence class of $\neg p$ n-formulae. Denote the numbers of the tautologies and antilogies, p-formulae and n-formulae of length $n-1$ with $\top_n^\Gamma$, $\bot_n^\Gamma$, P_n^Γ and N_n^Γ, respectively. If there is no danger of confusion, we omit the Γ from notations.

If we negate an n-formula, we get a p-formula, if we connect with a conjunction an n-formula and a p-formula we get an antilogy, etc. We summarize this in the following tables:

	$\neg$
$\bot$	$\top$
p	$\neg p$
$\neg p$	p
$\top$	$\bot$

$\wedge$	$\bot$	p	$\neg p$	$\top$
$\bot$	$\bot$	$\bot$	$\bot$	$\bot$
p	$\bot$	p	$\bot$	p
$\neg p$	$\bot$	$\bot$	$\neg p$	$\neg p$
$\top$	$\bot$	p	$\neg p$	$\top$

$\vee$	$\bot$	p	$\neg p$	$\top$
$\bot$	$\bot$	p	$\neg p$	$\top$
p	p	p	$\top$	$\top$
$\neg p$	$\neg p$	$\top$	$\neg p$	$\top$
$\top$	$\top$	$\top$	$\top$	$\top$

(2)

$\uparrow$	$\bot$	p	$\neg p$	$\top$
$\bot$	$\top$	$\top$	$\top$	$\top$
p	$\top$	$\neg p$	$\top$	$\neg p$
$\neg p$	$\top$	$\top$	p	p
$\top$	$\top$	$\neg p$	p	$\bot$

$\downarrow$	$\bot$	p	$\neg p$	$\top$
$\bot$	$\top$	$\neg p$	p	$\bot$
p	$\neg p$	$\neg p$	$\bot$	$\bot$
$\neg p$	p	$\bot$	p	$\bot$
$\top$	$\bot$	$\bot$	$\bot$	$\bot$

(3)

Theorem 2. *$P_n^{\{\neg,\vee\}} = P_n^{\{\neg,\wedge\}}$, $N_n^{\{\neg,\vee\}} = N_n^{\{\neg,\wedge\}}$, $\top_n^{\{\neg,\vee\}} = \bot_n^{\{\neg,\wedge\}}$ and $\bot_n^{\{\neg,\vee\}} = \top_n^{\{\neg,\wedge\}}$.*

Remark 4. These equivalence classes form a complete lattice which should remind the reader to the Belnap's 4 valued system [1], but in our case there are no two natural ordering. Here the operations join and meet are the operation *or* and *and*, respectively. Fortunately we have only four equivalence classes and we can formulate the operation of $\uparrow$ (*nand*) as the join of the complementer set of the set of the operands, e.g. $p \uparrow \neg p = \top \vee \bot$, where the operands are p and $\neg p$, and their complementer set is $\{\top, \bot\}$. Similar statement holds for $\downarrow$ (*nor*): it is the meet of the complementer set of the set of the operands.

Proof. We prove Theorem 2 by induction on n. In the logic of negation and conjunction $P_0^{\{\neg,\wedge\}} = N_0^{\{\neg,\wedge\}} = \bot_0^{\{\neg,\wedge\}} = \top_0^{\{\neg,\wedge\}} = 0$, $P_1^{\{\neg,\wedge\}} = N_1^{\{\neg,\wedge\}} = \bot_1^{\{\neg,\wedge\}} = \top_1^{\{\neg,\wedge\}} = 0$, $N_2^{\{\neg,\wedge\}} = \bot_2^{\{\neg,\wedge\}} = \top_2^{\{\neg,\wedge\}} = 0$ and $P_2^{\{\neg,\wedge\}} = 1$. Similar facts hold for $\top_n^{\{\neg,\vee\}}$, $\bot_n^{\{\neg,\vee\}}$, $P_n^{\{\neg,\vee\}}$ and $N_n^{\{\neg,\vee\}}$ if $n \leq 2$. Hence the statement is true for $n = 0$, 1 and 2.

According to (2) the following relations hold:

$$
\begin{aligned}
P_n^{\{\neg,\vee\}} &= N_{n-1}^{\{\neg,\vee\}} + \sum_{i=1}^{n-1} P_i^{\{\neg,\vee\}} \left(2\bot_{n-i}^{\{\neg,\vee\}} + P_{n-i}^{\{\neg,\vee\}}\right) \\
P_n^{\{\neg,\wedge\}} &= N_{n-1}^{\{\neg,\wedge\}} + \sum_{i=1}^{n-1} P_i^{\{\neg,\wedge\}} \left(2\top_{n-i}^{\{\neg,\wedge\}} + P_{n-i}^{\{\neg,\wedge\}}\right) \\
N_n^{\{\neg,\vee\}} &= P_{n-1}^{\{\neg,\vee\}} + \sum_{i=1}^{n-1} N_i^{\{\neg,\vee\}} \left(2\bot_{n-i}^{\{\neg,\vee\}} + N_{n-i}^{\{\neg,\vee\}}\right) \\
N_n^{\{\neg,\wedge\}} &= P_{n-1}^{\{\neg,\wedge\}} + \sum_{i=1}^{n-1} N_i^{\{\neg,\wedge\}} \left(2\top_{n-i}^{\{\neg,\wedge\}} + N_{n-i}^{\{\neg,\wedge\}}\right) \\
\bot_n^{\{\neg,\vee\}} &= \top_{n-1}^{\{\neg,\vee\}} + \sum_{i=1}^{n-1} \bot_i^{\{\neg,\vee\}} \bot_{n-i}^{\{\neg,\vee\}} \\
\top_n^{\{\neg,\wedge\}} &= \bot_{n-1}^{\{\neg,\wedge\}} + \sum_{i=1}^{n-1} \top_i^{\{\neg,\wedge\}} \top_{n-i}^{\{\neg,\wedge\}} \\
\top_n^{\{\neg,\vee\}} &= \bot_{n-1}^{\{\neg,\vee\}} + \sum_{i=1}^{n-1} \top_i^{\{\neg,\vee\}} \left(2\bot_{n-i}^{\{\neg,\vee\}} + 2P_{n-i}^{\{\neg,\vee\}} + 2N_{n-i}^{\{\neg,\vee\}} + \top_{n-i}^{\{\neg,\vee\}}\right) \\
&\quad + \sum_{i=1}^{n-1} P_i^{\{\neg,\vee\}} N_{n-i}^{\{\neg,\vee\}} \\
\bot_n^{\{\neg,\wedge\}} &= \top_{n-1}^{\{\neg,\wedge\}} + \sum_{i=1}^{n-1} \bot_i^{\{\neg,\wedge\}} \left(2\top_{n-i}^{\{\neg,\wedge\}} + 2P_{n-i}^{\{\neg,\wedge\}} + 2N_{n-i}^{\{\neg,\wedge\}} + \bot_{n-i}^{\{\neg,\wedge\}}\right) \\
&\quad + \sum_{i=1}^{n-1} P_i^{\{\neg,\wedge\}} N_{n-i}^{\{\neg,\wedge\}}
\end{aligned}
$$

Assume that our hypothesis hold for all numbers less than n. We want to prove it for n. Let us examine the equations of $P_n^{\{\neg,\wedge\}}$ and $P_n^{\{\neg,\vee\}}$! By the hypothesis $N_{n-1}^{\{\neg,\wedge\}} = N_{n-1}^{\{\neg,\vee\}}$, $P_i^{\{\neg,\wedge\}} = P_i^{\{\neg,\vee\}}$ and $\top_{n-i}^{\{\neg,\wedge\}} = \bot_{n-i}^{\{\neg,\vee\}}$ for all $i < n$. The right sides of equations are equal, hence $P_n^{\{\neg,\wedge\}}$ and $P_n^{\{\neg,\vee\}}$ are equal, too. The verification of the equalities of the other pairs is left to the reader. □

A similar theorem holds for the logic of nand and the logic of nor:

Theorem 3. $P_n^{\{\downarrow\}} = P_n^{\{\uparrow\}}$, $N_n^{\{\downarrow\}} = N_n^{\{\uparrow\}}$, $\top_n^{\{\downarrow\}} = \bot_n^{\{\uparrow\}}$ *and* $\bot_n^{\{\downarrow\}} = \top_n^{\{\uparrow\}}$.

Proof. By (3) we have:

$$
\begin{aligned}
P_n^{\{\uparrow\}} &= \sum_{i=1}^{n-1} N_i^{\{\uparrow\}} \left(2\top_{n-i}^{\{\uparrow\}} + N_{n-i}^{\{\uparrow\}}\right) \\
P_n^{\{\downarrow\}} &= \sum_{i=1}^{n-1} N_i^{\{\downarrow\}} \left(2\bot_{n-i}^{\{\downarrow\}} + N_{n-i}^{\{\downarrow\}}\right) \\
N_n^{\{\uparrow\}} &= \sum_{i=1}^{n-1} P_i^{\{\uparrow\}} \left(2\top_{n-i}^{\{\uparrow\}} + P_{n-i}^{\{\uparrow\}}\right) \\
N_n^{\{\downarrow\}} &= \sum_{i=1}^{n-1} P_i^{\{\downarrow\}} \left(2\bot_{n-i}^{\{\downarrow\}} + P_{n-i}^{\{\downarrow\}}\right) \\
\bot_n^{\{\uparrow\}} &= \sum_{i=1}^{n-1} \top_i^{\{\uparrow\}} \top_{n-i}^{\{\uparrow\}} \\
\top_n^{\{\downarrow\}} &= \sum_{i=1}^{n-1} \bot_i^{\{\downarrow\}} \bot_{n-i}^{\{\downarrow\}} \\
\top_n^{\{\uparrow\}} &= \sum_{i=1}^{n-1} \bot_i^{\{\uparrow\}} \left(2\top_{n-i}^{\{\uparrow\}} + 2P_{n-i}^{\{\uparrow\}} + 2N_{n-i}^{\{\uparrow\}} + \bot_{n-i}^{\{\uparrow\}}\right) \\
\bot_n^{\{\downarrow\}} &= \sum_{i=1}^{n-1} \top_i^{\{\downarrow\}} \left(2\bot_{n-i}^{\{\downarrow\}} + 2P_{n-i}^{\{\downarrow\}} + 2N_{n-i}^{\{\downarrow\}} + \top_{n-i}^{\{\downarrow\}}\right)
\end{aligned}
$$

The statement follows by induction on n. □

Theorem 4. *Each formula in $\mathcal{F}^{\{\uparrow\}}$ have odd length.*

Proof. We prove the statement by induction on the length of formulae of $\mathcal{F}^{\{\uparrow\}}$. The only formula of length 1 is p. Hence we can assume the statement for all formula of length less than or equal to n, where $n \geq 1$. Take a formula of length $n+1$, which can be only a formula of the form $A \uparrow B$, where the length of A and B are less than n, therefore both of them are odd-length formulae. By definition, the length of $A \uparrow B$ will be odd again, which concludes the proof. □

To simplify the calculations we construct four sets of equivalence classes: $a = \{\bot, p\}$, $b = \{\neg p, \top\}$, $c = \{\bot, \neg p\}$ and $d = \{p, \top\}$. Denote by A_n^Γ, B_n^Γ, C_n^Γ and by D_n^Γ the numbers of formulae length $n-1$ belonging to the sets a, b, c and d, respectively. It is obvious, that $A_n^\Gamma = \bot_n^\Gamma + P_n^\Gamma$, $B_n^\Gamma = \top_n^\Gamma + N_n^\Gamma$, $C_n^\Gamma = \bot_n^\Gamma + N_n^\Gamma$ and $D_n^\Gamma = \top_n^\Gamma + P_n^\Gamma$.

We leave it to the reader to check that if each operand is from the same set then the results of the operation also belong to the same set. We summarize this

in the following tables:

	$\neg$
a	b
b	a

	$\neg$
c	d
d	c

$\wedge$	a	b
a	a	a
b	a	b

$\wedge$	c	d
c	c	c
d	c	d

$\uparrow$	a	b
a	b	b
b	b	a

$\uparrow$	c	d
c	d	d
d	d	c

(4)

4 Generating functions

Let S_n^{Γ} denote the number of formulae of length $n-1$ of the logic corresponding to Γ if $n > 1$, and let $S_1 = S_0 = 0$.

Lemma 1.

$$S_n^{\{\neg,\wedge\}} = S_{n-1}^{\{\neg,\wedge\}} + \sum_{i=1}^{n-1} S_i^{\{\neg,\wedge\}} S_{n-i}^{\{\neg,\wedge\}} \text{ for } n > 2 \tag{5}$$

Proof. A formula of length $n-1$ can be either a negation of a formula of length $n-2$ or can be a formula of the kind of $\varphi \triangle \psi$, where φ and ψ has length $i-1$ and $n-1-i$, respectively. □

Let

$$S(x) = \sum_{n=0}^{\infty} S_n^{\{\neg,\wedge\}} x^n \tag{6}$$

be the generating function of $S_n^{\{\neg,\wedge\}}$. For the sake of brevity the generating function of $S_n^{\{\neg,\wedge\}}$ will be denoted by S.

Lemma 2.

$$S = \frac{1 - x - \sqrt{1 - 2x - 3x^2}}{2}$$

Proof. By (5) and the properties of generating functions we may write

$$S = xS + S^2 + x^2. \tag{7}$$

We had to add x^2 to the right side, because $S_2^{\{\neg,\wedge\}} \neq S_1^{\{\neg,\wedge\}} + S_1^{\{\neg,\wedge\}} S_1^{\{\neg,\wedge\}}$. Solving (7) in S we obtain two solutions, but one of them does not satisfy the condition $S(0) = S_0^{\{\neg,\wedge\}} = 0$, hence we have proved our lemma. □

By theorem 4 we have no even length formulae in the logic of nand, so to eliminate the zero values, we will use a bit different notation: $S_n^{\{\uparrow\}}$ denotes the number of formulae of length $2n-1$. It is easy to check that in this logic $S_1^{\{\uparrow\}} = 1$, moreover we set $S_0^{\{\uparrow\}} = 0$. Let $\overline{S}(x) = \sum_{n=0}^{\infty} S_n^{\{\uparrow\}} x^n$ be the generating function of $S_n^{\{\uparrow\}}$.

Lemma 3.

$$\overline{S} = \frac{1 - \sqrt{1 - 4x}}{2}$$

Proof. $S_n^{\{\uparrow\}} = \sum_{i=1}^{n-1} S_i^{\{\uparrow\}} S_{n-i}^{\{\uparrow\}}$, because $2n-1 = 2i-1+2(n-i)-1+1$. Therefore $\overline{S} = \overline{S}\overline{S}+x$, and solving this by using $\overline{S}(0) = S_0^{\{\uparrow\}} = 0$ we proved our statement. □

5 Logic of negation and conjunction

Remark 5. By Theorem 2 the relative frequencies of the logic of negation and disjunction can be derived easily from the relative frequencies of the logic of negation and conjunction.

In this section the set Γ is $\{\neg, \wedge\}$. For the sake of brevity we will omit Γ, so write $\bot_n$ instead $\bot_n^{\{\neg,\wedge\}}$, etc.

Lemma 4. *Let B, D and P be the generating functions of B_n, D_n and P_n, respectively. Then*

$$B = \frac{1 + x - \sqrt{1 + 3x^2 + 2xY}}{2} \tag{8}$$

$$D = \frac{1 + x - Z}{2} \tag{9}$$

$$P = \frac{2x - Z + \sqrt{7x^2 - 2xZ + 1 + 2xY - 2x\sqrt{1 + 3x^2 + 2xY}}}{2} \tag{10}$$

Here $Y = \sqrt{-(x+1)(3x-1)}$ and $Z = \sqrt{1 - x^2 + 2xY}$.

Proof. By (4),

$$B_n = A_{n-1} + \sum_{i=1}^{n-1} B_i B_{n-i}$$

that is we obtain a tautology or a n-formula of length $n-1$ by negating an antilogy or a p-formula of length $n-2$, or by conjugating two suitably long tautologies or n-formulae. Since $S_n = A_n + B_n = C_n + D_n$, thus $B = x(S - B) + B^2$ holds. Solving it and using that $B(0) = B_0 = 0$, we obtain (8). Similarly, we have $D = x(S - D) + D^2 + x^2$, using that $D(0) = D_0 = 0$, we achieve (9). We obtain a p-formula of length $n-1$ either by negating an n-formula of length $n-2$, or by conjugating two suitably long p-formulae or a p-formula and a tautology. Hence,

$$P_n = N_{n-1} + \sum_{i=1}^{n-1} P_i P_{n-i} + \sum_{i=1}^{n-1} \top_i P_{n-i} + \sum_{i=1}^{n-1} P_i \top_{n-i}.$$

Since $\top_j = D_j - P_j$, and $N_j = C_j - (A_j - P_j)$, we have $P = xC + xA + xP + P^2 + 2P(D - P) + x^2$. Solving this equation and using that $P(0) = P_0 = 0$, we obtain (10). □

Lemma 5. *The singularity of S, B, D and P (defined by (6) and in lemma 4, respectively) with minimal modulus is* $1/3$.

Proof. By Remark 1, the functions S, B, D and P may have singularities when

$$1 - 2x - 3x^2 = 0 \tag{11}$$

$$1 + 3x^2 + 2x\sqrt{1 - 2x - 3x^2} = 0 \tag{12}$$

$$1 - x^2 + 2x\sqrt{1 - 2x - 3x^2} = 0 \tag{13}$$

or

$$\begin{aligned} 7x^2 &- 2x\sqrt{1 - x^2 + 2xY} + 1 + 2x\sqrt{1 - 2x - 3x^2} \\ &- 2x\sqrt{1 + 3x^2 + 2x\sqrt{1 - 2x - 3x^2}} = 0 \end{aligned}$$

The roots of (11) are -1 and $1/3$. The equation (12) has two solutions, which are approximately $-0.4110 - 0.3592i$ and $-0.4110 + 0.3592i$ having modulus ≈ 0.5458. The equation (13) has also two solutions, which are -1 and ≈ -0.3733. Finally the last equation has two solutions, which are approximately $0.6434 + 0.3378i$ and $0.6434 - 0.3378i$ having modulus ≈ 0.7267.

Among the possible singularities $1/3$ has the smallest modulus, and $1/3$ is a singularity for all of S, B, D and P. □

Theorem 5. *The relative frequency of tautologies, p-formulae, n-formulae and antilogies of length n are asymptotically the following:*

$$\lim_{n\to\infty} \frac{P_n}{S_n} = \frac{\sqrt{2}}{4} - \frac{12-3\sqrt{2}-2\sqrt{3}}{24\sqrt{4-\sqrt{3}-\sqrt{2}}} \approx 0.15995$$

$$\lim_{n\to\infty} \frac{\top_n}{S_n} = \frac{12-3\sqrt{2}-2\sqrt{3}}{24\sqrt{4-\sqrt{3}-\sqrt{2}}} \approx 0.19360$$

$$\lim_{n\to\infty} \frac{N_n}{S_n} = \frac{\sqrt{3}}{6} - \frac{12-3\sqrt{2}-2\sqrt{3}}{24\sqrt{4-\sqrt{3}-\sqrt{2}}} \approx 0.09507$$

$$\lim_{n\to\infty} \frac{\bot_n}{S_n} = \frac{12-3\sqrt{2}-2\sqrt{3}}{24}\left(2 - \frac{1}{\sqrt{4-\sqrt{3}-\sqrt{2}}}\right) \approx 0.55138$$

Proof. By Lemma 5, the generating function $S(x)$ of the sequence S_n has the minimal singularity $1/3$, and the function

$$S(x/3) = \frac{3 - x - \sqrt{9 - 6x - 3x^2}}{6}$$

has an expansion in the form $\frac{1}{3} - \frac{\sqrt{3}}{3}\sqrt{1-x} + O(1-x)$. Again by Lemma 5, the generating function $P(x)$ of the sequence P_n has a singularity at $1/3$, which is the one with the least modulus, and the function $P(x/3)$ has an expansion

$$\frac{1 + \sqrt{4 - \sqrt{3} - \sqrt{2}} - \sqrt{2}}{3} + \left(\frac{4\sqrt{3} - \sqrt{6} - 2}{24\sqrt{4 - \sqrt{3} - \sqrt{2}}} - \frac{\sqrt{6}}{12}\right)\sqrt{1-x} + O(1-x)\ ,$$

whence by Theorem 1,

$$\lim_{n\to\infty} \frac{P_n}{S_n} = \frac{\sqrt{2}}{4} - \frac{12 - 3\sqrt{2} - 2\sqrt{3}}{24\sqrt{4 - \sqrt{3} - \sqrt{2}}} \ .$$

Similarly, the generating function $D(x)$ of the sequence D_n has singularity $1/3$ with minimal modulus, and the function $D(x/3)$ has an expansion $\frac{2-\sqrt{2}}{3} - \frac{\sqrt{6}}{12}\sqrt{1-x} + O(1-x)$. Hence by the definition of D_n and Theorem 1,

$$\begin{aligned}\lim_{n\to\infty} \frac{\top_n}{S_n} &= \lim_{n\to\infty} \frac{D_n - P_n}{S_n} = \lim_{n\to\infty} \frac{D_n}{S_n} - \lim_{n\to\infty} \frac{P_n}{S_n} = \\ &\frac{\sqrt{2}}{4} - \left(\frac{\sqrt{2}}{4} - \frac{12 - 3\sqrt{2} - 2\sqrt{3}}{24\sqrt{4 - \sqrt{3} - \sqrt{2}}}\right) = \\ &\frac{12 - 3\sqrt{2} - 2\sqrt{3}}{24\sqrt{4 - \sqrt{3} - \sqrt{2}}} \ .\end{aligned}$$

Furthermore, the generating function $B(x)$ of the sequence B_n has singularity $1/3$ with minimal modulus, and the function $B(x/3)$ has an expansion $\frac{2-\sqrt{3}}{3} - \frac{1}{6}\sqrt{1-x} + O(1-x)$. Hence by the definition of B_n and Theorem 1,

$$\begin{aligned}\lim_{n\to\infty} \frac{N_n}{S_n} &= \lim_{n\to\infty} \frac{B_n - \top_n}{S_n} = \lim_{n\to\infty} \frac{B_n}{S_n} - \lim_{n\to\infty} \frac{\top_n}{S_n} = \\ &\frac{\sqrt{3}}{6} - \frac{12 - 3\sqrt{2} - 2\sqrt{3}}{24\sqrt{4 - \sqrt{3} - \sqrt{2}}} \ .\end{aligned}$$

Since $S_n = A_n + B_n$, thus

$$\begin{aligned}\lim_{n\to\infty} \frac{\bot_n}{S_n} &= \lim_{n\to\infty} \frac{A_n - P_n}{S_n} = \lim_{n\to\infty} \frac{S_n - B_n - P_n}{S_n} = \\ &\lim_{n\to\infty} \frac{S_n}{S_n} - \lim_{n\to\infty} \frac{B_n}{S_n} - \lim_{n\to\infty} \frac{P_n}{S_n} = \\ &1 - \frac{\sqrt{3}}{6} - \left(\frac{\sqrt{2}}{4} - \frac{12 - 3\sqrt{2} - 2\sqrt{3}}{24\sqrt{4 - \sqrt{3} - \sqrt{2}}}\right) = \\ &\frac{12 - 3\sqrt{2} - 2\sqrt{3}}{24}\left(2 - \frac{1}{\sqrt{4 - \sqrt{3} - \sqrt{2}}}\right) \ .\end{aligned}$$

□

6 Logic of nand

Remark 6. By Theorem 3 the relative frequencies of the logic of nor can be derived easily from the relative frequencies of the logic of nand.

In this section the set Γ is $\{\uparrow\}$. For the sake of brevity we omit Γ, hence we write $\perp_n$ instead $\perp_n^{\{\uparrow\}}$, etc.

Lemma 6. *Let A, C and P be the generating functions of A_n, C_n and P_n, respectively. Then*

$$A = 1 - \frac{1}{2}Y - \frac{1}{2}Z_2 \tag{14}$$

$$C = 1 - \frac{1}{2}Y - \frac{1}{2}Z_1 \tag{15}$$

$$P = -1 + \frac{1}{2}Z_1 + \frac{1}{2}\sqrt{7 - 2Z_1 - 2Y - 2Z_2} \tag{16}$$

where $Y = \sqrt{1-4x}$, $Z_1 = \sqrt{3-2Y}$ and $Z_2 = \sqrt{3-2Y-4x}$.

Proof. By (4),

$$A_n = \sum_{i=1}^{n-1}(S_i - A_i)(S_{n-i} - A_{n-i}) \quad \text{and} \quad C_n = \sum_{i=1}^{n-1}(S_i - C_i)(S_{n-i} - C_{n-i}).$$

For the corresponding generating functions we have $A = (\overline{S} - A)(\overline{S} - A) + x$ and $C = (\overline{S} - C)(\overline{S} - C)$. Solving these equations we get (14) and (15). By (3) and the definition of A, B, C and D, we have

$$P_n = \sum_{i=1}^{n-1} N_i(2\top_{n-i} + N_{n-i}) = \sum_{i=1}^{n-1} N_i\,(2B_{n-i} - N_{n-i}) = \sum_{i=1}^{n-1}(C_i - A_i + P_i)\,(2S_{n-i} - A_{n-i} - C_{n-i} - P_{n-i}))\ .$$

From this we get $P = (C - A + P)(2\overline{S} - A - C - P) + x$. Solving this equation we get (16). □

Lemma 7. *The singularity of $\overline{S}$, A, C and P (defined in lemma 6) with minimal modulus is* $1/4$.

Proof. The functions $\overline{S}$, A, C and P may have singularities when

$$1 - 4x = 0 \tag{17}$$

$$3 - 2\sqrt{1-4x} = 0 \tag{18}$$

$$3 - 2\sqrt{1-4x} - 4x = 0 \tag{19}$$

or

$$7 - 2\sqrt{3 - 2\sqrt{1-4x}} - 2\sqrt{1-4x} - 2\sqrt{3 - 2\sqrt{1-4x} - 4x} = 0.$$

The root of (17) is $1/4$. The equation (18) has solution $-5/16$, and the equation (19) has two solutions: $1/4 - 1/2i$ and $1/4 + 1/2i$. Finally the last equation has two

solutions, which are approximately $-0.41823 - 0.48428i$ and $-0.41823 + 0.48428i$ having modulus ≈ 0.63987.

Among the possible singularities $1/4$ has the smallest modulus, and $1/4$ is a singularity for all of $\overline{S}$, A, C and P. □

Theorem 6. *The relative frequency of tautologies, p-formulae, n-formulae and antilogies of length n are asymptotically the following:*

$$\begin{aligned}
\lim_{n\to\infty} \frac{P_n}{S_n} &= \frac{\sqrt{3}}{3} - \frac{2\sqrt{3}-6+3\sqrt{2}}{6\sqrt{7-2\sqrt{3}-2\sqrt{2}}} && \approx 0.23916\\
\lim_{n\to\infty} \frac{\top_n}{S_n} &= \frac{2\sqrt{3}-6+3\sqrt{2}}{6\sqrt{7-2\sqrt{3}-2\sqrt{2}}} && \approx 0.33819\\
\lim_{n\to\infty} \frac{N_n}{S_n} &= \frac{\sqrt{2}}{2} + \frac{2\sqrt{3}-6+3\sqrt{2}}{6\sqrt{7-2\sqrt{3}-2\sqrt{2}}} && \approx 0.36892\\
\lim_{n\to\infty} \frac{\bot_n}{S_n} &= \frac{6-2\sqrt{3}-3\sqrt{2}}{6}\left(1 + \frac{1}{\sqrt{7-2\sqrt{3}-2\sqrt{2}}}\right) && \approx 0.05373
\end{aligned}$$

Proof. By Lemma 7, the generating function $\overline{S}(x)$ of the sequence S_n has the only singularity $1/4$, and the function

$$\overline{S}(x/4) = \frac{1-\sqrt{1-x}}{2}$$

has an expansion in the form $\frac{1}{2} - \frac{1}{2}\sqrt{1-x}$. Again by Lemma 7, the generating function $P(x)$ of the sequence P_n has a singularity at $1/4$, which is the one with the least modulus, and the function $P(x/4)$ has an expansion $\frac{-2+\sqrt{3}+\sqrt{7-2\sqrt{3}-2\sqrt{2}}}{2} + \left(-\frac{\sqrt{3}}{6} + \frac{2\sqrt{3}-6+3\sqrt{2}}{12\sqrt{7-2\sqrt{3}-2\sqrt{2}}}\right)\sqrt{1-x} + O(1-x)$,whence by Theorem 1,

$$\lim_{n\to\infty} \frac{P_n}{S_n} = \frac{\sqrt{3}}{3} - \frac{2\sqrt{3}-6+3\sqrt{2}}{6\sqrt{7-2\sqrt{3}-2\sqrt{2}}}\,.$$

Similarly, the generating function $A(x)$ of the sequence A_n has singularity $1/4$ with minimal modulus, and the function $A(x/4)$ has an expansion $1 - \frac{\sqrt{2}}{2} + \frac{-2+\sqrt{2}}{4}\sqrt{1-x} + O(1-x)$. Hence by the definition of A_n and Theorem 1,

$$\begin{aligned}
\lim_{n\to\infty} \frac{\bot_n}{S_n} &= \lim_{n\to\infty} \frac{A_n - P_n}{S_n} = 1 - \frac{\sqrt{2}}{2} - \left(\frac{\sqrt{3}}{3} - \frac{2\sqrt{3}-6+3\sqrt{2}}{6\sqrt{7-2\sqrt{3}-2\sqrt{2}}}\right) =\\
&\frac{6-2\sqrt{3}-3\sqrt{2}}{6}\left(1 + \frac{1}{\sqrt{7-2\sqrt{3}-2\sqrt{2}}}\right)\,.
\end{aligned}$$

Furthermore, the generating function $C(x)$ of the sequence C_n has singularity $1/4$ with minimal modulus, and the function $C(x/4)$ has an expansion $1 - \frac{\sqrt{3}}{2} +$

$\frac{-3+\sqrt{3}}{6}\sqrt{1-x} + O(1-x)$. Hence by the definition of C_n and Theorem 1,

$$\lim_{n\to\infty} \frac{N_n}{S_n} = \lim_{n\to\infty} \frac{C_n - \bot_n}{S_n} = \frac{\sqrt{2}}{2} + \frac{2\sqrt{3} - 6 + 3\sqrt{2}}{6\sqrt{7 - 2\sqrt{3} - 2\sqrt{2}}} .$$

Since $S_n = C_n + D_n$, thus

$$\lim_{n\to\infty} \frac{\top_n}{S_n} = \lim_{n\to\infty} \frac{D_n - P_n}{S_n} = \frac{2\sqrt{3} - 6 + 3\sqrt{2}}{6\sqrt{7 - 2\sqrt{3} - 2\sqrt{2}}} .$$

□

7 Summary

The logic of negation and equivalence is not a known logic, because in the general case we cannot express any truth-value function with the help of negation and equivalence. But if we have only one propositional variable, we can construct formulae equivalent with p, $\neg p$, $\top$ and $\bot$. We leave it to the reader to check that a formula of this language is a tautology if and only if it contains even number of negations and odd number of equivalences. (Hint. Use the following rewrite rules: $(\neg\varphi \equiv \psi) \sim \neg(\varphi \equiv \psi)$, $((\varphi \equiv \psi) \equiv \chi) \sim \varphi \equiv \psi \equiv \chi$, $p \equiv p \equiv p \sim p$, $\varphi \equiv \psi \sim \psi \equiv \varphi$ and double negation.)

From this it follows that the length of each tautology is even. This is the reason why $\top_n/S_n$ has no limit. We will discuss this logic in a separate article.

Following the method described in the proof of Theorem 1, we were able compute the relative frequency of tautologies, p-formulae, n-formulae and antilogies in several logics, determined by different combinations of connectives. The table below summarizes the numerical results of the calculations of different ratios.

Γ	Approximation of proportion of			
	p-formulae	antilogies	n-formulae	tautologies
$\uparrow$	23.916%	5.373%	36.892%	33.819%
$\downarrow$	23.916%	33.819%	36.892%	5.373%
$\neg\wedge$	15.995%	55.138%	9.507%	19.360%
$\neg\vee$	15.995%	19.360%	9.507%	55.138%
$\neg\wedge\vee$	32.498%	26.081%	15.341%	26.081%
$\neg\wedge\supset$	27.803%	19.986%	15.906%	36.305%
$\neg\wedge\supset\equiv$	27.804%	20.681%	17.787%	33.729%

References

[1] Belnap, Jr., Nuel D. A useful four-valued logic. In *Modern uses of multiple-valued logic (Fifth Internat. Sympos., Indiana Univ., Bloomington, Ind., 1975)*, pages 5–37. Episteme, Vol. 2. Reidel, Dordrecht, 1977.

[2] Harris, Mitchell A. Counting satisfiable k-cnf formulas. In Walsh, Toby, editor, *Principles and Practice of Constraint Programming - CP 2001, 7th International Conference, CP 2001, Paphos, Cyprus, November 26 - December 1, 2001, Proceedings*, volume 2239 of *Lecture Notes in Computer Science*, page 765. Springer, 2001.

[3] Kostrzycka, Zofia. On the density of truth in Grzegorczyk's modal logic. *Bull. Sect. Logic Univ. Łódź*, 33(2):107–120, 2004.

[4] Kostrzycka, Zofia. Density of truth in modal logics. In *Fourth Colloquium on Mathematics and Computer Science Algorithms, Trees, Combinatorics and Probabilities, Proceedings*, pages 161–170. www.dmtcs.org, 2006.

[5] Kostrzycka, Zofia and Zaionc, Marek. Statistics of intuitionistic versus classical logics. *Studia Logica*, 76(3):307–328, 2004.

[6] Moczurad, M., Tyszkiewicz, J., and Zaionc, M. Statistical properties of simple types. *Math. Structures Comput. Sci.*, 10(5):575–594, 2000.

[7] Soza, Grzegorz. Asymptotic density as a method of expressing quantitative relations in intuitionistic logic. *Rep. Math. Logic*, (36):71–93, 2002.

[8] Szegő, Gábor. *Orthogonal polynomials*, volume XXIII of *AMS Colloquium Publications*. AMS, 1939.

[9] Wilf, Herbert S. *generatingfunctionology*. A K Peters, third edition, 2005.

[10] Zaionc, Marek. On the asymptotic density of tautologies in logic of implication and negation. *Rep. Math. Logic*, (39):67–87, 2005.

[11] Zaionc, Marek. Probability distribution for simple tautologies. *Theor. Comput. Sci.*, 355(2):243–260, 2006.

Received 8th September 2009

Acta Cybernetica **20** (2012) 399–417.

Approximation of the Euclidean Distance by Chamfer Distances*

András Hajdu†, Lajos Hajdu‡, and Robert Tijdeman§

Abstract

Chamfer distances play an important role in the theory of distance transforms. Though the determination of the exact Euclidean distance transform is also a well investigated area, the classical chamfering method based upon "small" neighborhoods still outperforms it e.g. in terms of computation time. In this paper we determine the best possible maximum relative error of chamfer distances under various boundary conditions. In each case some best approximating sequences are explicitly given. Further, because of possible practical interest, we give all best approximating sequences in case of small (i.e. 5×5 and 7×7) neighborhoods.

Keywords: Chamfering, Approximation of the Euclidean distance, Distance transform, Digital image processing

1 Introduction

Suppose we measure distances between grid points of a two-dimensional grid and we want to approximate the Euclidean distance by a distance function which can be computed quickly, without calculating square roots. We may then use the class of chamfer distances. They are obtained by prescribing the lengths of the grid vectors in a so-called mask $M_p := \{(x, y) \in \mathbb{Z}^2 : \max(|x|, |y|) \leq p\}$ (for some positive integer

*Research of the Hungarian authors was supported in part by the OTKA grants F043090, T042985, T048791, K67580, K75566, NK101680, by the János Bolyai Research Fellowship of the Hungarian Academy of Sciences, by the TECH08-2 project DRSCREEN - Developing a computer based image processing system for diabetic retinopathy screening of the National Office for Research and Technology of Hungary (contract no.: OM-00194/2008, OM-00195/2008, OM-00196/2008), and by the TÁMOP 4.2.1./B-09/1/KONV-2010-0007 project, which is implemented through the New Hungary Development Plan, cofinanced by the European Social Fund and the European Regional Development Fund.

†Faculty of Informatics, University of Debrecen, H-4010 Debrecen, P.O. Box 12, Hungary. E-mail: hajdu.andras@inf.unideb.hu

‡Number Theory Research Group of the Hungarian Academy of Sciences and Institute of Mathematics, University of Debrecen, H-4010 Debrecen, P.O. Box 12, Hungary. E-mail: hajdul@math.unideb.hu

§Mathematical Institute, Leiden University. Niels Bohrweg 1, Leiden Postbus 9512, 2300 RA Leiden, The Netherlands. E-mail: tijdeman@math.leidenuniv.nl

p) such that the values at $(\pm x, \pm y)$ and $(\pm y, \pm x)$ are all the same, and by defining the length function W as follows: the length $W(\vec{v})$ of any vector $\vec{v} \in \mathbb{Z}^2$ is defined as the minimal sum of the lengths of those vectors from M_p, repetitions permitted, which have sum $\vec{v}$. The literature on chamfer distances is very rich. See Borgefors [2, 3, 4] for the basics, [17, 18] for lists of $(2p+1) \times (2p+1)$ neighborhoods for $1 \leq p \leq 10$, and [7] for an overview of applications. Further, recently many related results have been obtained by several authors, concerning distance transforms and their explicit calculation using different kinds of neighborhoods in certain (mostly 3D) grids. For example, Strand, Nagy, Fouard and Borgefors [20] gave a sequential algorithm for computing the distance map using distances based on neighborhood sequences in the 2D square grid, and 3D cubic and so-called FCC and BCC cubic grids, respectively. Similar results for other kinds of grids are also known, see e.g. [16] (nD hexagonal grids), [15] (diamond grid) and [11] (general point grids) and the references given there.

Classical chamfer distances using 3×3, 5×5 and 7×7 neighborhoods given by Borgefors [2, 3] are generated by the masks

$$\begin{array}{ccc} & & \\ \underline{4} & \underline{3} & \underline{4} \\ \underline{3} & 0 & \underline{3} \\ \underline{4} & \underline{3} & \underline{4} \end{array}, \quad \begin{array}{ccccc} & & & & \\ & \underline{11} & 10 & \underline{11} & \\ \underline{11} & \underline{7} & \underline{5} & \underline{7} & \underline{11} \\ 10 & \underline{5} & 0 & \underline{5} & 10 \\ \underline{11} & \underline{7} & \underline{5} & \underline{7} & \underline{11} \\ & \underline{11} & 10 & \underline{11} & \end{array} \quad \text{and} \quad \begin{array}{ccccccc} & & & & & & \\ & \underline{43} & \underline{38} & 36 & \underline{38} & \underline{43} & \\ \underline{43} & 34 & \underline{27} & 24 & \underline{27} & 34 & \underline{43} \\ \underline{38} & \underline{27} & \underline{17} & \underline{12} & \underline{17} & \underline{27} & \underline{38} \\ 36 & 24 & \underline{12} & 0 & \underline{12} & 24 & 36 \\ \underline{38} & \underline{27} & \underline{17} & \underline{12} & \underline{17} & \underline{27} & \underline{38} \\ \underline{43} & 34 & \underline{27} & 24 & \underline{27} & 34 & \underline{43} \\ & \underline{43} & \underline{38} & 36 & \underline{38} & \underline{43} & \end{array}$$

respectively (with the actual generator entries underlined). For comparison with the Euclidean distance the values of the neighborhoods have to be divided by 3, 5 and 12, respectively. The approximations to $\sqrt{2} \approx 1.41$ are therefore $4/3 \approx 1.33$, $7/5 = 1.4$ and $17/12 \approx 1.41$, respectively. For alternative neighborhood values see Verwer [22, 23], Thiel [21], Coquin and Bolon [6], Butt and Maragos [5] and Scholtus [17]. More specifically, in [6] the minimization of the error between the Euclidean distance and the local distance was considered over circular trajectories similarly to [22, 23] rather than linear ones [3, 21]. The approximation error can also be measured based on area as it is done in [5] with calculating the difference between a disk of large size obtained by chamfer metric and a Euclidean disk of the same radius. The determination of the exact Euclidean distance transform is also a well investigated area (see e.g. [1, 7, 8, 13, 19]), but the classical 3×3 chamfering method still outperforms it in terms of computation time and simple extendability to other grids.

In this paper we determine chamfer distances best approximating the Euclidean distance in a certain sense. In each neighborhood size some best approximating sequences are explicitly given. Further, because of possible practical interest, we give all best approximating sequences in case of small (i.e. 5×5 and 7×7) neighborhoods.

Throughout the paper, as a measure for the quality of a length function W

defined on $\mathbb{Z}^2$ we use the so-called maximum relative error (m.r.error for short)

$$E := \limsup_{|\vec{v}|\to\infty} \left| \frac{W(\vec{v})}{|\vec{v}|} - 1 \right|$$

where $|.|$ denotes the Euclidean length. The M_1-, M_2- and M_3-neighborhoods given above yield rounded E-values 0.0572, 0.0198 and 0.0138, respectively. Firstly we shall prove that the smallest possible constant E_p^B for the mask M_p under the condition that $W(x,0) = |x|$ for $x \in \mathbb{Z}$ is given by

$$E_p^B = \frac{p^2 + 2 - p\sqrt{p^2+1} - 2\sqrt{p^2+1-p\sqrt{p^2+1}}}{p^2} = \frac{1.5-\sqrt{2}}{p^2} + O\left(\frac{1}{p^4}\right).$$

In particular, $E_1^B \approx 0.0551$, $E_2^B \approx 0.0187$ and $E_3^B \approx 0.0089$. Comparing these values with the E-values given above, one can see that the E_p^B-values yield approximately 4%, 6% and 35% improvement, respectively. The B refers to Borgefors who was the first to consider such neighborhoods.

Secondly we consider the case D in which $W(\vec{v}) \geq |\vec{v}|$ for all $\vec{v} \in \mathbb{Z}^2$. (The D refers to the fact that $W(\vec{v})$ dominates $|\vec{v}|$.) The optimal m.r.error under this restriction equals

$$E_p^D = \sqrt{(\sqrt{p^2+1}-p)^2+1} - 1 = \frac{1}{8p^2} + O\left(\frac{1}{p^4}\right) = \frac{0.125}{p^2} + O\left(\frac{1}{p^4}\right).$$

In particular, $E_1^D \approx 0.0824$, $E_2^D \approx 0.0275$ and $E_3^D \approx 0.0131$.

Thirdly we shall prove that the optimal E-value without any restriction on the neighborhood defined on M_p (i.e. dropping the condition $W(x,0) = |x|$ for $x \in \mathbb{Z}$) equals

$$E_p^C = \frac{\sqrt{2p^2+2-2p\sqrt{p^2+1}} - 1}{\sqrt{2p^2+2-2p\sqrt{p^2+1}} + 1} = \frac{1}{16p^2} + O\left(\frac{1}{p^4}\right).$$

In particular, $E_1^C \approx 0.0396$, $E_2^C \approx 0.0136$ and $E_3^C \approx 0.0065$. In 1991, on using the symmetry in case C the value of E_p^C was computed by Verwer [22, 23] in terms of trigonometric functions. The C refers to the word central. In 1998, because of geometric considerations, Butt and Maragos [5] chose to use the error function

$$\limsup_{|\vec{v}|\to\infty} \left| \frac{|\vec{v}|}{W(\vec{v})} - 1 \right|$$

which of course is small if and only if E_p^C is small. In general it gives different error values, but the values for E_p^C are equal to the values obtained by the above error function (cf. Scholtus [17]). We prove the correctness of the above E_p^C values. In doing so, our motivation is twofold: on the one hand, by a simple reasoning we obtain these values immediately from the values of E_p^D, and on the other hand, our

proofs are mathematically rigorous while the corresponding arguments of Verwer and Butt and Maragos contain some hidden assumptions. Namely, by certain plausible but not explicitly verified geometric arguments they restrict their attention and investigations to certain values of the neighborhoods in question, and they perform exact investigations only for these values.

We shall further study an auxiliary class of neighborhoods on M_p, viz. the class of neighborhoods satisfying $\mathcal{N}_c(\vec{v}) = \infty$ for all $\vec{v} = (x, y) \in M_p$ with either $x < p$ or $y < 0$, $\mathcal{N}_c(\vec{v}) = p$ for $\vec{v} = (p, 0)$, and $\mathcal{N}_c(\vec{v}) = c|\vec{v}|$ for $\vec{v} = (p, k)$ with $0 < k \leq p$. Here c is a constant close to and at most equal to 1. Informally speaking, the use of such neighborhoods means that only such steps (v_1, v_2) are allowed, where v_1 is a positive multiple of p and v_2 is nonnegative. Further, beside $\mathcal{N}_c(p, 0) = p$ the weights of the other such neighborhood vectors are their Euclidean lengths, multiplied by a factor $c \leq 1$. All the other vectors of the neighborhood are forbidden to use, thus they have weights ∞. For example, the weights for the neighborhood $\mathcal{N}_c$ with $p = 2$ (i.e. for M_2) are given by

$$\begin{array}{ccccc} \infty & \infty & \infty & \infty & c\sqrt{8} \\ \infty & \infty & \infty & \infty & c\sqrt{5} \\ \infty & \infty & \infty & \infty & 2 \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{array}$$

where the origin is in the middle. We denote the maximum relative error for this class of neighborhoods by $\mathcal{E}_p^c$ where we restrict the limsup to vectors $\vec{v}$ with finite lengths $W(\vec{v})$ (i.e. having coordiantes (x, y) with $0 \leq y \leq x$ and $p \mid x$). Our motivation for considering such neighborhoods is that it will turn out that (due to its special form) $\mathcal{N}_c$ is easier to handle, but yields the same m.r.error as the corresponding neighborhood N_c, in which $N_c(\pm p, 0) = N_c(0, \pm p) = p$ and $N_c(x, y) = c\sqrt{x^2 + y^2}$ otherwise ($(x, y) \in M_p$).

In Section 2 we introduce some notation and prove some preliminary results. In Sections 3 and 4 we compute the values of $\mathcal{E}_p^B$ and $\mathcal{E}_p^D$ where $\mathcal{E}_p^B$ is the maximum relative error $\mathcal{E}_p^c$ for optimal c and $\mathcal{E}_p^D = \mathcal{E}_p^1$. We give all sequences yielding minimal m.r.error in case of 5×5 and 7×7 neighborhoods, as well. In Section 4 we prove that $E_p^B = \mathcal{E}_p^B$ and $E_p^D = \mathcal{E}_p^D$ and further show that $E_p^C = E_p^D/(2 + E_p^D)$ for all p. Finally, we draw some conclusions in Section 5.

2 Definitions and basic properties

Let N be a neighborhood defined on the mask M_p. Put $M_p^* = M_p \setminus \{(0, 0)\}$. We denote the value of N at position (n, k) by $w(n, k)$ for $(n, k) \in M_p$. Throughout the paper we assume that $w(\pm n, \pm k) = w(\pm k, \pm n) > 0$ for all $(n, k) \in M_p^*$ and all possible sign choices. Hence it suffices to consider the values $w(n, k)$ with $0 \leq k \leq n \leq p$.

We can measure lengths of vectors and distances between points using neighborhood sequences. Note that such sequences provide a flexible and very useful

tool in handling several problems in discrete geometry. For the basics and most important facts about such sequences, see e.g. the papers [9, 24, 10, 12, 14] and the references given there. Here we only give those notions which will be needed for our purposes.

Let $A = (N_i)_{i=1}^{\infty}$ be a sequence of neighborhoods defined on M_p and $\vec{u}, \vec{v} \in \mathbb{Z}^2$. The sequence $\vec{u} = \vec{u}_0, \vec{u}_1, \ldots, \vec{u}_m = \vec{v}$ with $\vec{u}_i - \vec{u}_{i-1} \in M_p$ is called an A-path from $\vec{u}$ to $\vec{v}$. The A-length of the path is defined as $\sum_{i=1}^{m} w_i(\vec{u}_i - \vec{u}_{i-1})$. The distance $W_A(\vec{v}-\vec{u})$ between $\vec{u}$ and $\vec{v}$, which is the A-length of $\vec{v}-\vec{u}$, is defined as the minimal A-length taken over all A-paths from $\vec{u}$ to $\vec{v}$. If the neighborhood sequence is fixed, then we suppress the letter A in the above notation:

If $N_i = N$ for all i, then the corresponding (constant) neighborhood sequence is denoted by $A = \overline{N}$. We assume throughout the paper that for such sequences $W(n,k) = w(n,k)$ holds for $(n,k) \in M_p$; if it would not have been the case, then the function $w := W|_{M_p^*}$ would have generated W, too.

We call W a metric if for all $\vec{u}, \vec{v} \in \mathbb{Z}^2$

- $W(\vec{u}) < \infty$ (W is finite),
- $W(\vec{u}) = 0 \Leftrightarrow \vec{u} = \vec{0}$ (W is positive definite),
- $W(\vec{u}) = W(-\vec{u})$ (W is symmetric),
- $W(\vec{u} + \vec{v}) \leq W(\vec{u}) + W(\vec{v})$ (W satisfies the triangle inequality).

It follows from the above properties that $W(\vec{u}) \geq 0$ for every $\vec{u} \in \mathbb{Z}^2$. By our basic assumptions on w, every induced length function W is positive definite and symmetric. Furthermore, W satisfies the triangle inequality for $\vec{u}, \vec{v}$ with $\vec{u}, \vec{v}, \vec{u}+\vec{v} \in M_p$ by definition.

The first lemma shows that for a constant neighborhood sequence $W(\vec{v})/|\vec{v}|$ attains a minimal value which is reached already in M_p^*.

Lemma 1. *Let N be a neighborhood defined on M_p which induces the length function W on $\mathbb{Z}^2$. Then*

$$\liminf_{|\vec{v}|\to\infty} \frac{W(\vec{v})}{|\vec{v}|} = \min_{\vec{v}\in M_p^*} \frac{w(\vec{v})}{|\vec{v}|}.$$

Proof. Let $m = \min_{\vec{v}\in M_p^*} \frac{w(\vec{v})}{|\vec{v}|} = \frac{w(\vec{u})}{|\vec{u}|}$ ($\vec{u} \in M_p^*$). Then for all n we have $\frac{W(n\vec{u})}{|n\vec{u}|} = m$, so that $\liminf_{|\vec{v}|\to\infty} \frac{W(\vec{v})}{|\vec{v}|} \leq m$. On the other hand, since $\frac{w(\vec{v})}{|\vec{v}|} \geq m$ for every $\vec{v} \in M_p^*$, it follows from the definition of shortest path and the triangle inequality for the Euclidean distance that

$$W(\vec{v}) \geq \sum_i w(\vec{v}_i) = \sum_i \frac{w(\vec{v}_i)}{|\vec{v}_i|} \cdot |\vec{v}_i| \geq m \sum_i |\vec{v}_i| \geq m|\vec{v}|$$

for every $\vec{v} \in \mathbb{Z}^2$ not equal to the origin. Thus $\liminf_{|\vec{v}|\to\infty} \frac{W(\vec{v})}{|\vec{v}|} \geq m$.

□

The challenge is therefore to compute $\limsup_{|\vec{v}|\to\infty} \frac{W(\vec{v})}{|\vec{v}|}$.

3 The maximum relative error for neighborhoods $\mathcal{N}_c$

Let c be some positive real number with $\frac{p}{\sqrt{p^2+1}} < c \leq 1$. We shall study neighborhoods $\mathcal{N}_c$ on M_p with $\mathcal{N}_c(n,k) = \infty$ for which either $n < p$ or $k < 0$, $\mathcal{N}_c(p,0) = p$ and $\mathcal{N}_c(p,k) = c\sqrt{p^2+k^2}$ for $0 < k \leq p$. We are interested in the length function $\mathcal{W}_c$ induced by $\mathcal{A}_c := \overline{\mathcal{N}_c}$ for points in the set $\{(x,y) \in \mathbb{Z}^2 : p|x, 0 \leq y \leq x\}$. First we secure that under suitable conditions only two distinct steps occur in a shortest $\mathcal{A}_c$-path.

Lemma 2. *Let* $\frac{p}{\sqrt{p^2+1}} < c \leq 1$. *Then a shortest* $\mathcal{A}_c$*-path from* $(0,0)$ *to* $(mp, mr+k)$ *with* $m, r, k \in \mathbb{Z}, 0 \leq r < p, 0 \leq k < m$ *consists only of steps* (p,r) *and* $(p,r+1)$.

Proof. Suppose a shortest path from $(0,0)$ to $(mp, mr+k)$ with $m, r, k \in \mathbb{Z}, 0 \leq r < p, 0 \leq k < m$ contains two steps (p,t) and (p,u) with $t - 2 \geq u \geq 0$. Replace the two steps with steps $(p, t-1)$ and $(p, u+1)$, and write L_1 and L_2 for the length of the old and new paths, respectively. Then we have

$$L_1 - L_2 \geq c\sqrt{p^2+t^2} - c\sqrt{p^2+(t-1)^2} + c\sqrt{p^2+u^2} - c\sqrt{p^2+(u+1)^2} =$$

$$= c(f_p(t) - f_p(u+1)),$$

where

$$f_p(x) = \sqrt{p^2+x^2} - \sqrt{p^2+(x-1)^2} \quad (x \in \mathbb{Z}_{\geq 0}).$$

A simple calculation yields that $f_p(x)$ is strictly monotone increasing in x, which shows that $L_1 - L_2 > 0$. However, this contradicts the minimality of the length of the original path.

Hence a shortest path may contain steps (p,t) and $(p,t+1)$ only, for some non-negative integer t. Since altogether we make m steps, this immediately gives that $t = r$, and our statement follows.

□

Remark 1. The latter inequality is the most severe and explains why we restrict c to values greater than $\frac{p}{\sqrt{p^2+1}}$.

Corollary 1. *Let* $\frac{p}{\sqrt{p^2+1}} < c \leq 1$ *Then a shortest* $\mathcal{A}_c$*-path from* $(0,0)$ *to* (mp, mr) *with* $0 \leq r \leq p$ *consists of* m *steps* (p,r).

The next theorem gives the value of the approximation error for general p, in case of any neighborhood $\mathcal{N}_c$ on M_p.

Theorem 1. *Let* $p \geq 1$, $\frac{p}{\sqrt{p^2+1}} < c \leq 1$. *Then the m.r.error of* $\mathcal{A}_c$ *to the Euclidean distance is given by*

$$\max(1-c, \sqrt{1+c^2+p^2+c^2p^2-2cp\sqrt{p^2+1}}-1).$$

Proof. As a general remark we mention that to perform our calculations, we used the program package Maple ®[1].

Let p be a positive integer, and fix c with $\frac{p}{\sqrt{p^2+1}} < c \leq 1$. As previously, it is sufficient to consider the $\mathcal{A}_c$-length of points of the form (mp, k) where m is some positive integer and k is an integer with $0 \leq k \leq mp$. Write $k = mq + r$ with $0 \leq q \leq p$ and $0 \leq r < m$. The possible steps are $(p, 0)$ of length p and $(p, \pm i)$ of length $W_i := c\sqrt{p^2+i^2}$ (for $|i| \leq p$). From Lemma 2 and the inequalities $p = W_0 < W_1 < \ldots < W_p$ we see that a path of minimal length from $(0,0)$ to a point $(mp, mq+r)$ consists of r steps $(p, q+1)$ and $m-r$ steps (p, q). Hence for the induced length function we get

$$\mathcal{W}(mp, mq+r) = rW_{q+1} + (m-r)W_q.$$

Put $t = r/m$, and recall that $W_0 = p$ and $W_i = c\sqrt{p^2+i^2}$ for $i = 1, \ldots, p$. Set

$$H_0(t) = \frac{ct\sqrt{p^2+1} + p(1-t)}{\sqrt{p^2+t^2}} - 1,$$

and for $1 \leq q \leq p$

$$H_q(t) = c\frac{t\sqrt{p^2+(q+1)^2} + (1-t)\sqrt{p^2+q^2}}{\sqrt{p^2+(q+t)^2}} - 1,$$

and let

$$h_q(p,c) = \max_{0 \leq t \leq 1} |H_q(t)| \quad (0 \leq q < p) \quad \text{and} \quad h_p(p,c) = |H_p(0)|.$$

Now we investigate the error functions $h_q(p,c)$ for $q = p, q = 0, 0 < q < p$, respectively.

Suppose first that $q = p$. Then $r = 0$ and $k = mp$. In this case we trivially have $h_p(p,c) = 1 - c$.

Assume next that $q = 0$. Then $0 \leq k < p$. Put

$$t_0 := p(c\sqrt{p^2+1} - p).$$

A simple calculation yields that $0 \leq t_0 \leq 1$, and that H_0 is monotone increasing on the interval $[0, t_0]$ and monotone decreasing on the interval $[t_0, 1]$. Moreover, we have $H_0(0) = 0$ and $H_0(1) = c - 1$, hence $H_0(t_0) \geq 0$. Thus we have

$$h_0(p,c) = \max(1-c, H_0(t_0)) = \max(1-c, \sqrt{1+c^2+p^2+c^2p^2-2cp\sqrt{p^2+1}}-1).$$

[1]Maple is a registered trademark of Waterloo Maple Inc.

Finally, suppose that $0 < q < p$, that is $p \le k < mp$. Put

$$t_q := \frac{\sqrt{p^2+q^2}(\sqrt{(p^2+q^2)(p^2+(q+1)^2)} - p^2 - q^2 - q)}{(q+1)\sqrt{p^2+q^2} - q\sqrt{p^2+(q+1)^2}}.$$

A simple calculation gives that $0 \le t_q \le 1$, and that H_q is monotone increasing on the interval $[0, t_q]$, while monotone decreasing on the interval $[t_q, 1]$. We also have $H_q(0) = H_q(1) = c - 1$. Hence $H_q(t_q) < 0$ implies $-H_q(t_q) \le 1 - c$. Thus we get

$$h_q(p,c) = \max(1-c, H_q(t_q)) = \max\left(1-c, c\sqrt{\frac{2}{1+\sqrt{1-\frac{p^2}{(p^2+q^2)(p^2+(q+1)^2)}}}} - 1\right).$$

Now we calculate the error function

$$h(p,c) := \limsup_{p|n,\ n \ge k \ge 0} \left| \frac{W(n,k)}{\sqrt{n^2+k^2}} - 1 \right| = \max_{0 \le q \le p} h_q(p,c).$$

Observe first that for fixed p and c the function $h_q(p,c)$ is monotone decreasing in q with $2 \le q \le p$. Hence $h_q(p,c) \le h_1(p,c)$ for $q = 2, \ldots, p$. Further, again by Maple, we obtain that for any c with $\frac{p}{\sqrt{p^2+1}} < c \le 1$

$$c\sqrt{\frac{2}{1+\sqrt{1-\frac{p^2}{(p^2+1)(p^2+4)}}}} \le \sqrt{1+c^2+p^2+c^2p^2-2cp\sqrt{p^2+1}}$$

holds, which implies $h_1(p,c) \le h_0(p,c)$. Hence

$$h(p,c) = \max(1-c, \sqrt{1+c^2+p^2+c^2p^2-2cp\sqrt{p^2+1}} - 1)$$

and the theorem follows.

□

The following corollaries provide the m.r.errors $\mathcal{E}_p^B$ (when $c = c_p^B$) and $\mathcal{E}_p^D$ (when $c = 1$), respectively.

Corollary 2. *Let p be a positive integer. Then we have*

$$c_p^B = \frac{p\sqrt{p^2+1} + 2\sqrt{p^2+1-p\sqrt{p^2+1}} - 2}{p^2}.$$

That is, the sequence $\mathcal{A} = \mathcal{A}_{c_p^B}$ of period p given by $\mathcal{A} = \overline{N_{c_p^B}}$ yields the smallest m.r.error among all sequences $\mathcal{A}_c$ of period p. Moreover, the error is given by

$$\mathcal{E}_p^B = 1 - c_p^B = \frac{p^2+2-p\sqrt{p^2+1}-2\sqrt{p^2+1-p\sqrt{p^2+1}}}{p^2}$$

$$= \frac{1.5-\sqrt{2}}{p^2} + O\left(\frac{1}{p^4}\right) \approx \frac{0.0858}{p^2} + O\left(\frac{1}{p^4}\right).$$

Proof. Put

$$f(c) = 1 - c \text{ and } g(c) = \sqrt{1 + c^2 + p^2 + c^2p^2 - 2cp\sqrt{p^2+1}} - 1.$$

A straightforward computation shows that f is strictly monotone decreasing, while g is strictly monotone increasing for $\frac{p}{\sqrt{p^2+1}} < c \leq 1$. Hence there is a unique solution of the equation $f(c) = g(c)$ in this interval. By Theorem 1 this solution is given by

$$c_p^B = \frac{p\sqrt{p^2+1} + 2\sqrt{p^2+1-p\sqrt{p^2+1}} - 2}{p^2}.$$

Thus the statement follows.

□

Corollary 3. *Let p be a positive integer. Then the sequence $\mathcal{A} = \mathcal{A}_1$ of period p given by $\mathcal{A} = \overline{N_1}$ (corresponding to the choice $c = 1$) has m.r.error*

$$\mathcal{E}_p^D = \sqrt{(\sqrt{p^2+1} - p)^2 + 1} - 1 = \frac{1}{8p^2} + O\left(\frac{1}{p^4}\right) = \frac{0.125}{p^2} + O\left(\frac{1}{p^4}\right).$$

Proof. On substituting $c = 1$ into the formula of Theorem 1, the statement follows immediately.

□

Now we give the best approximating sequences realizing the minimal maximum relative error for 5×5 matrices ($p = 2$) in Theorem 2 and for 7×7 matrices ($p = 3$) in Theorem 3, respectively.

Theorem 2. *Let $\frac{2}{\sqrt{5}} < c \leq 1$. Let $\mathcal{A}_c = \overline{N_c}$ be the corresponding sequence on M_2. Then the minimal m.r.error to the Euclidean distance among the neighborhood sequences $\mathcal{A}_c$ is attained if and only if*

$$c = c_2^B, \quad W_1 = s \quad \text{and} \quad u \leq W_2 \leq v,$$

where

$$s = \frac{5 - \sqrt{5} + \sqrt{25 - 10\sqrt{5}}}{2} \approx 2.1943,$$

$$u = \frac{2\sqrt{2}}{\sqrt{5}} s \approx 2.7756 \quad \text{and} \quad v = 2 + \frac{2s}{5} \approx 2.8777.$$

Further, the m.r.error is given by

$$\mathcal{E}_2^B = 1 - c_2^B = 1 - \frac{s}{\sqrt{5}} = \frac{3 - \sqrt{5} - \sqrt{5 - 2\sqrt{5}}}{2} \approx 0.0187.$$

Proof. For any even n with $0 \le k \le n$ the possible steps are $(2,0)$ of length 2, $(2,1)$ and $(2,-1)$ of length W_1, and $(2,2)$ and $(2,-2)$ of length W_2. From Lemma 2 and the inequality $2 < W_1 < W_2$ we see that the path from $(0,0)$ to (n,k) of minimal length consists of k steps $(2,1)$ and $\frac{n}{2}-k$ steps $(2,0)$ if $0 \le k \le n/2$ and of $k-n/2$ steps $(2,2)$ and $n-k$ steps $(2,1)$ if $\frac{n}{2} \le k \le n$. Hence we have for the induced length function

$$\mathcal{W}(n,k) = \begin{cases} kW_1 + n - 2k, & \text{if } k \le \frac{n}{2}, \\ (n-k)W_1 + (k-\frac{n}{2})W_2, & \text{otherwise.} \end{cases}$$

Put $t = k/n$. Then the error function is given by

$$h(W_1,W_2) := \limsup_{2|n,\ n \ge k \ge 0} \left| \frac{\mathcal{W}(n,k)}{\sqrt{n^2+k^2}} - 1 \right| =$$

$$\max\left(\max_{0 \le t \le \frac{1}{2}} \left| \frac{t(W_1-2)+1}{\sqrt{1+t^2}} - 1 \right|, \max_{\frac{1}{2} \le t \le 1} \left| \frac{(1-t)W_1 + (t-\frac{1}{2})W_2}{\sqrt{1+t^2}} - 1 \right| \right).$$

Our aim is to choose W_1 and W_2 such that $h(W_1,W_2)$ is minimal. For fixed W_1, define the function $H_0 : \mathbb{R}_{\ge 0} \to \mathbb{R}$ by

$$H_0(t) = \frac{t(W_1-2)+1}{\sqrt{1+t^2}}.$$

Put $t_0 = W_1 - 2$. We observe that H_0 is monotone increasing on $[0,t_0]$ and monotone decreasing on $[t_0,\infty)$. Hence, as $H_0(0) = 1$,

$$\max_{0 \le t \le \frac{1}{2}} (|H_0(t) - 1|) = \max\left(H_0(t_0) - 1, 1 - H_0\left(\frac{1}{2}\right) \right)$$

$$= \max\left(\sqrt{W_1^2 - 4W_1 + 5} - 1, 1 - \frac{W_1}{\sqrt{5}} \right)$$

if $W_1 \le 5/2$ and

$$\max_{0 \le t \le \frac{1}{2}} (|H_0(t) - 1|) = H_0\left(\frac{1}{2}\right) - 1 = \frac{W_1}{\sqrt{5}} - 1$$

otherwise. Clearly,

$$\min_{W_1,W_2} (h(W_1,W_2)) \ge \min_{W_1} \max\left(\sqrt{W_1^2 - 4W_1 + 5} - 1, \left| 1 - \frac{W_1}{\sqrt{5}} \right| \right). \tag{1}$$

A calculation gives that the minimum of the right-hand side is achieved for

$$W_1 = s := \frac{5 - \sqrt{5} + \sqrt{25 - 10\sqrt{5}}}{2} \approx 2.1943$$

and equals

$$\sqrt{s^2-4s+5}-1=1-\frac{s}{\sqrt{5}}=\frac{3-\sqrt{5}-\sqrt{5-2\sqrt{5}}}{2}\approx 0.0187.$$

Now we fix the value s of W_1, and show that we can choose W_2 in a way to have equality in (1). In fact we completely describe the set of the appropriate W_2-s. Consider the maximum over $t\in[1/2,1]$. For fixed W_2, define the function $H_1:\ \mathbb{R}_{\geq 0}\to\mathbb{R}$ by

$$H_1(t)=\frac{(1-t)W_1+(t-\frac{1}{2})W_2}{\sqrt{1+t^2}}.$$

Observe that H_1 attains its maximum at $t_1:=\frac{2(W_2-W_1)}{2W_1-W_2}$ (which is positive) and further, H_1 is monotone increasing in $[0,t_1]$ and monotone decreasing in $[t_1,\infty)$. Hence

$$\max_{\frac{1}{2}\leq t\leq 1}(|H_1(t)-1|)=\max\left(1-H_1\left(\frac{1}{2}\right),H_1(t_1)-1,1-H_1(1)\right)=$$

$$=\max\left(1-\frac{W_1}{\sqrt{5}},\frac{\sqrt{(2W_1-W_2)^2+4(W_2-W_1)^2}}{2}-1,1-\frac{W_2}{2\sqrt{2}}\right)$$

if $1/2\leq t_1\leq 1$, and

$$\max_{\frac{1}{2}\leq t\leq 1}(|H_1(t)-1|)=\max\left(|H_1\left(\frac{1}{2}\right)-1|,|H_1(1)-1|\right)=$$

$$=\max\left(1-\frac{W_1}{\sqrt{5}},\left|\frac{W_2}{2\sqrt{2}}-1\right|\right)$$

otherwise. By our choice of W_1, we have that

$$1-\frac{W_1}{\sqrt{5}}=\left|1-\frac{s}{\sqrt{5}}\right|\approx 0.0187.$$

The values of $\left|\frac{W_2}{2\sqrt{2}}-1\right|$ and $\left|\frac{\sqrt{(2W_1-W_2)^2+4(W_2-W_1)^2}}{2}-1\right|$ do not exceed this value if and only if $u\leq W_2\leq v$ where u and v are defined in the statement of the theorem. We conclude that $h(W_1,W_2)$ attains its minimum $1-\frac{s}{\sqrt{5}}$ if $W_1=s$ and $u\leq W_2\leq v$.

The above argument shows that $\mathcal{E}_2^B=1-\frac{W_1}{\sqrt{5}}$. Hence the minimum among neighborhoods $\mathcal{N}_c$ is realized for $c=c_2^B=\frac{W_1}{\sqrt{5}}$ and for no other value of c.

□

Theorem 3. *Let* $\frac{3}{\sqrt{10}}<c\leq 1$. *Let* $\mathcal{A}_c=\overline{\mathcal{N}_c}$ *be the corresponding sequence on* M_3. *Then the minimal m.r.error to the Euclidean distance among the neighborhood sequences* $\mathcal{A}_c$ *is attained if and only if*

$$c=c_3^B,\quad W_1=s,\ u\leq W_2\leq v,\ q\leq W_3\leq r,$$

where

$$s = \frac{30 - 2\sqrt{10} + 2\sqrt{100 - 30\sqrt{10}}}{9} \approx 3.1340,$$

$$u = \frac{\sqrt{13}}{\sqrt{10}} s \approx 3.5733,$$

$$v = \frac{143s - 18\sqrt{13} + 6\sqrt{1690 - 143\sqrt{13}s}}{121} \approx 3.5944,$$

$$q = \frac{3s}{\sqrt{5}} \approx 4.2047,$$

$$r = \frac{3\sqrt{13s^2 - 52\sqrt{10}s + 520 - 10W_2^2}}{13\sqrt{10}} + \frac{15W_2}{13},$$

and in the definition of r, W_2 *can be any number with* $u \le W_2 \le v$. *Further, the m.r.error is given by*

$$\mathcal{E}_3^B = 1 - c_3^B = 1 - \frac{s}{\sqrt{10}} = \frac{11 - 3\sqrt{10} - 2\sqrt{10 - 3\sqrt{10}}}{9} \approx 0.0089.$$

Proof. Let $3|n$ and $0 \le k \le n$. The possible steps are $(3,0)$ of length 3, $(3,\pm 1)$ of length W_1, $(3,\pm 2)$ of length W_2, and $(3,\pm 3)$ of length W_3. From the inequalities $3 < W_1 < W_2 < W_3$ it follows that the path from $(0,0)$ to (n,k) of minimal length consists of k steps $(3,1)$ and $\frac{n}{3} - k$ steps $(3,0)$ if $0 \le k \le \frac{n}{3}$; of $k - \frac{n}{3}$ steps $(3,2)$ and $\frac{2n}{3} - k$ steps $(3,1)$ if $\frac{n}{3} \le k \le \frac{2n}{3}$; of $k - \frac{2n}{3}$ steps $(3,3)$ and $n - k$ steps $(3,2)$ if $\frac{2n}{3} \le k \le n$. Hence we have for the induced length function

$$\mathcal{W}(n,k) = \begin{cases} kW_1 + n - 3k, & \text{if } k \le n/3, \\ (2n/3 - k)W_1 + (k - n/3)W_2, & \text{if } n/3 < k \le 2n/3, \\ (n-k)W_2 + (k - 2n/3)W_3, & \text{otherwise.} \end{cases}$$

Put $t = k/n$, and define the functions $H_i : \mathbb{R}_{\ge 0} \to \mathbb{R}$ $(i = 0,1,2)$ by

$$H_0(t) = \frac{t(W_1 - 3) + 1}{\sqrt{1+t^2}}, \qquad H_1(t) = \frac{\left(\frac{2}{3} - t\right) W_1 + \left(t - \frac{1}{3}\right) W_2}{\sqrt{1+t^2}}$$

and

$$H_2(t) = \frac{(1-t)W_2 + \left(t - \frac{2}{3}\right) W_3}{\sqrt{1+t^2}}.$$

Then for fixed W_1, W_2, W_3 the error of approximation is given by

$$h(W_1, W_2, W_3) = \max\left(\max_{0 \le t \le \frac{1}{3}} |H_0(t) - 1|, \max_{\frac{1}{3} \le t \le \frac{2}{3}} |H_1(t) - 1|, \max_{\frac{2}{3} \le t \le 1} |H_2(t) - 1| \right).$$

Let

$$t_0 = W_1 - 3, \; t_1 = \frac{3(W_2 - W_1)}{2W_1 - W_2}, \; t_2 = \frac{3(W_3 - W_2)}{3W_2 - 2W_3},$$

and observe that all t_0, t_1 and t_2 are positive. By differentiation and following standard calculus, we get that for $i = 0, 1, 2$, H_i is monotone decreasing if $t_i \notin [i/3, (i+1)/3]$, and that H_i is monotone increasing in $[i/3, t_i]$ and monotone decreasing in $[t_i, (i+1)/3]$ otherwise. Hence from $H_0(0) = 1$ we get that

$$\max_{0 \le t \le \frac{1}{3}} (|H_0(t) - 1|) = \max\left(H_0(t_0) - 1, \left|H_0\left(\frac{1}{3}\right) - 1\right|\right) =$$

$$= \max\left(\sqrt{W_1^2 - 6W_1 + 10} - 1, \left|\frac{W_1}{\sqrt{10}} - 1\right|\right).$$

Hence obviously,

$$\min_{W_1, W_2, W_3} h(W_1, W_2, W_3) \ge \min_{W_1} \max\left(\sqrt{W_1^2 - 6W_1 + 10} - 1, \left|1 - \frac{W_1}{\sqrt{10}}\right|\right). \quad (2)$$

By a simple calculation we get that the minimum of the right-hand side is achieved for

$$W_1 = s := \frac{30 - 2\sqrt{10} + 2\sqrt{100 - 30\sqrt{10}}}{9} \approx 3.1340$$

and equals

$$M := \sqrt{s^2 - 6s + 10} - 1 = 1 - \frac{s}{\sqrt{10}} = \frac{3\sqrt{10} + 7 + 2\sqrt{10 - 3\sqrt{10}}}{9} \approx 0.0089.$$

Now we fix the value s of W_1, and show that we can choose W_2 and W_3 in a way to have equality in (2). More precisely, we completely describe the set of the appropriate pairs (W_2, W_3). For this purpose, first we consider the maximum of H_1 over $t \in [1/3, 2/3]$. In a similar manner as in the proof of Theorem 2, we obtain that

$$\max_{\frac{1}{3} \le t \le \frac{2}{3}} (|H_1(t) - 1|) = \max\left(\left|H_1\left(\frac{1}{3}\right) - 1\right|, H_1(t_1) - 1, \left|H_1\left(\frac{2}{3}\right) - 1\right|\right) =$$

$$= \max\left(\left|\frac{W_1}{\sqrt{10}} - 1\right|, \frac{\sqrt{(2W_1 - W_2)^2 + 9(W_2 - W_1)^2}}{3} - 1, \left|\frac{W_2}{\sqrt{13}} - 1\right|\right).$$

Using our choice for W_1, a simple calculation gives that the above maximum does not exceed the value of M precisely when $u \le W_2 \le v$, where u and v are defined in the statement of the theorem. So let W_2 be any fixed number from the interval $[u, v]$, and consider the the maximum of H_2 over $t \in [2/3, 1]$. Now we get that

$$\max_{\frac{2}{3} \le t \le 1} (|H_2(t) - 1|) = \max\left(\left|H_2\left(\frac{2}{3}\right) - 1\right|, H_2(t_2) - 1, |H_2(1) - 1|\right) =$$

$$= \max\left(\left|\frac{W_2}{\sqrt{13}} - 1\right|, \frac{\sqrt{(3W_2 - 2W_3)^2 + 9(W_3 - W_2)^2}}{3} - 1, \left|\frac{W_3}{3\sqrt{2}} - 1\right|\right).$$

Using our choice for W_1 and W_2, a simple calculation yields that the above maximum is not larger than M if and only if $q \leq W_3 \leq r$, where q and r are given in the statement. (Note that $4.2766 < r < 4.2804$.)

The above argument shows that $\mathcal{E}_3^B = 1 - \frac{W_1}{\sqrt{10}}$. Hence the minimum among neighborhoods $\mathcal{N}_c$ is realized for $c = c_3^B = \frac{W_1}{\sqrt{10}}$, and the theorem follows.

□

4 Equivalence of m.r.errors for M_p neighborhoods

In this section we compute the m.r.errors E_p^B, E_p^C and E_p^D. First we introduce neighborhoods N_c on M_p defined by $N_c(0,0) = \infty, N_c(n,0) = N_c(0,n) = |n|$ for $0 < |n| \leq p$, $N_c(n,k) = c\sqrt{n^2+k^2}$ for $(n,k) \in M_p, nk \neq 0$. Let W_c denote the length function induced by the sequence $\overline{N_c}$. We show that the corresponding m.r.error E_c satisfies $E_c = \mathcal{E}_c$ for every considered value of c. It then follows that $E_p^B = \mathcal{E}_p^B$ and $E_p^D = \mathcal{E}_p^D$ for every $p \geq 1$.

Lemma 3. *Let* $\frac{p}{\sqrt{p^2+1}} < c \leq 1$. *There is a shortest* $\overline{N_c}$*-path from* $(0,0)$ *to* (mp,k) *with* $0 \leq k \leq m$ *which consists of steps of the form* $(p,0)$ *and* $(p,1)$.

Proof. Suppose a shortest path from $(0,0)$ to (mp,k) contains a step (g,h) with $h < 0$. Then it also contains a step (i,j) with $j \geq 1$. But it is shorter to replace both steps with steps $(g,h+1)$ and $(i,j-1)$. A similar argument can be used to exclude steps (g,h) with $h > 1$. So every shortest path from $(0,0)$ to (mp,k) contains only steps of the forms $(g,0)$ and $(g,1)$.

If $k = m$, then taking only steps $(p,1)$ gives the shortest path length because of the triangle inequality for the Euclidean distance and the inequality $c \leq 1$. Suppose that there is a step $(g,1)$ with $g < p$ in a shortest path from $(0,0)$ to (mp,k) with $0 \leq k < m$. Then there is also a step $(h,0)$ with $h > 0$. But we can replace both steps with steps $(g+1,1)$ and $(h-1,0)$ and make the path shorter. Therefore all the steps of the form $(g,1)$ are of the form $(p,1)$. The remaining steps can be combined to steps of the form $(p,0)$.

□

Lemma 4. *Let* p *be fixed. Let* $\frac{p}{\sqrt{p^2+1}} < c \leq 1$. *The m.r.error of the neighborhood sequence* $\overline{N_c}$ *is equal to* $\mathcal{E}_p^D$ *if* $c = 1$ *and equal to* $\mathcal{E}_p^B$ *if* c *assumes the value* c_p^B *from Corollary 2.*

Proof. Because of symmetry it suffices only to consider points (n,k) with $0 \leq k \leq n$. First let $c = 1$. By definition $N(n,k) = \sqrt{n^2+k^2}$ for $(n,k) \in M_p$. Hence the induced length function satisfies $W_1(n,k) \geq |(n,k)|$ for all $(n,k) \in \mathbb{Z}^2$. Thus

$$\min \frac{W_1(n,k)}{\sqrt{n^2+k^2}} \geq 1$$

where the minimum is taken over all $(n,k) \in \mathbb{Z}^2$ with $(n,k) \neq (0,0)$. On the other hand, by Lemma 3, the shortest $\overline{N_1}$ path from $(0,0)$ to (mp,k) with $0 \leq k \leq m$ consists of steps of the forms $(p,0)$ and $(p,1)$ which have lengths p and $\sqrt{p^2+1}$, respectively. Hence $W_1(mp,k) = \mathcal{W}_1(mp,k)$ for $0 \leq k \leq m$. If $n = mp + r$ with $0 \leq r < p$, then $|W_1(n,k) - W_1(mp,k)| < p$. Note that in view the proof of Theorem 1 (in particular, since $h_0(p,c) \geq h_i(p,c)$ for all $i \geq 1$ there) we have

$$\limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq mp}} \frac{\mathcal{W}_1(mp,k)}{|(mp,k)|} = \limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq m}} \frac{\mathcal{W}_1(mp,k)}{|(mp,k)|}.$$

Thus on the one hand it follows that

$$\limsup_{\substack{|(n,k)|\to\infty \\ 0\leq k\leq n}} \frac{W_1(n,k)}{|(n,k)|} = \limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq mp}} \frac{W_1(mp,k)}{|(mp,k)|} \geq$$

$$\geq \limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq m}} \frac{W_1(mp,k)}{|(mp,k)|} = \limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq m}} \frac{\mathcal{W}_1(mp,k)}{|(mp,k)|} = \limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq mp}} \frac{\mathcal{W}_1(mp,k)}{|(mp,k)|}.$$

On the other hand, by $W_1(mp,k) \leq \mathcal{W}_1(mp,k)$ for all m,p and k, we also have that

$$\limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq mp}} \frac{\mathcal{W}_1(mp,k)}{|(mp,k)|} \geq \limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq mp}} \frac{W_1(mp,k)}{|(mp,k)|} = \limsup_{\substack{|(n,k)|\to\infty \\ 0\leq k\leq n}} \frac{W_1(n,k)}{|(n,k)|}.$$

Hence

$$\limsup_{\substack{|(n,k)|\to\infty \\ 0\leq k\leq n}} \frac{W_1(n,k)}{|(n,k)|} = \limsup_{\substack{|(mp,k)|\to\infty \\ 0\leq k\leq mp}} \frac{\mathcal{W}_1(mp,k)}{|(mp,k)|}$$

and by

$$\limsup_{|(mp,k)|\to\infty} \frac{\mathcal{W}_1(mp,k)}{|(mp,k)|} = 1 + \mathcal{E}_p^D,$$

the m.r.error of $\overline{N_1}$ equals $\mathcal{E}_p^D$.

Next let $c = c_p^B = 1 - \mathcal{E}_p^B$. Then $\frac{p}{\sqrt{p^2+1}} < c < 1$, and, by construction, $\mathcal{W}_c(p,0) = p, \mathcal{W}_c(p,k) = c\sqrt{p^2+k^2}$ for $0 < k \leq p$, and $W_c(n,k) = c\sqrt{n^2+k^2}$ for $0 < k \leq n \leq p$. Hence

$$\min_{(n,k)\in M_p^*} \frac{W_c(n,k)}{\sqrt{n^2+k^2}} = c = 1 - \mathcal{E}_p^B.$$

Thus

$$\liminf_{|(n,k)|\to\infty} \frac{W_{c_p^B}(n,k)}{|(n,k)|} = 1 - \mathcal{E}_p^B.$$

On the other hand, by Lemma 3, the shortest $\overline{N_c}$ path from $(0,0)$ to (mp,k) with $0 \le k \le m$ consists of steps of the form $(p,0)$ and $(p,1)$. By a similar reasoning as above we obtain that

$$\limsup_{|(n,k)|\to\infty} \frac{W_{c_p^B}(n,k)}{|(n,k)|} = 1 + \mathcal{E}_p^B.$$

Thus the m.r.error of $\overline{N_c}$ equals $\mathcal{E}_p^B$.

□

Theorem 4. *For every p we have $E_p^B = \mathcal{E}_p^B$ and $E_p^D = \mathcal{E}_p^D$.*

Proof. We first consider the D-case. Suppose the neighborhood N on M_p induces a length function $W : \mathbb{Z}^2 \to \mathbb{R}_{\ge 0}$ such that $W(\vec{v}) \ge |\vec{v}|$ for all $\vec{v} \in \mathbb{Z}^2$ and W has m.r.error E_p^D. It can only improve the m.r.error if we replace the value $N(n,k)$ for some $(n,k) \in M_p^*$ with a smaller value $\ge |(n,k)|$. Therefore we may assume without loss of generality that $N = N_1$. Hence $E_p^D = \mathcal{E}_p^D$.

Now we turn to the B-case. Suppose a neighborhood N on M_p induces a length function W such that $W(n,0) = W(0,n) = |n|$ for $n \in \mathbb{Z}$ and $(1-E_p^B)|\vec{v}| \le W(\vec{v}) \le (1+E_p^B)|\vec{v}|$ for all $\vec{v} \in \mathbb{Z}^2$. Without loss of generality we may replace all values $N(n,k)$ for $(n,k) \in M_p^*$ with $|n|$ if $k = 0$, with $|k|$ if $n = 0$, and with $(1-E_p^B)|(n,k)|$ otherwise. Thus E_p^B equals the m.r.error of the neighborhood sequence $\overline{N_{1-E_p^B}}$. We know from Lemma 4 and Corollary 2 that if $c = c_p^B$, then the m.r.error of $\overline{N_c}$ equals $\mathcal{E}_p^B = 1 - c_p^B$. Hence $E_p^B \le \mathcal{E}_p^B$. From $N(n,k) \ge (1-E_p^B)|(n,k)| \ge c_p^B|(n,k)|$ for all $(n,k) \in M_p^*$ we obtain $W(\vec{v}) \ge W_{c_p^B}(\vec{v})$ for all $\vec{v} \in \mathbb{Z}^2$. Hence

$$1 + E_p^B = \inf_N \limsup_{|\vec{v}|\to\infty} \frac{W(\vec{v})}{|\vec{v}|} \ge \limsup_{|\vec{v}|\to\infty} \frac{W_{c_p^B}(\vec{v})}{|\vec{v}|} = 1 + \mathcal{E}_p^B$$

by Lemma 4. Thus $E_p^B = \mathcal{E}_p^B$.

□

Finally, we compute the minimal m.r.error E_p^C for the class of arbitrary neighborhoods N defined on M_p. Observe that the m.r.error E_p^C is attained by the length function W corresponding to the neighborhood N defined by $w(\vec{v}) = (1-E_p^C)|\vec{v}|$ for $\vec{v} \in M_p^*$, since $\frac{N(\vec{v})}{|\vec{v}|}$ should not assume a smaller value than $1 - E_p^C$ and the limsup-value cannot increase if we decrease some $w(\vec{v})$. Clearly, the length function W corresponding to $\overline{N}$ is just $\frac{1}{1-E_p^C} W_1$ where W_1 is the length function on $\overline{N_1}$. Recall that $\overline{N_1}$ has m.r.error E_p^D. Therefore we have

$$1 + E_p^C = \limsup_{|\vec{v}|\to\infty} \frac{W(\vec{v})}{|\vec{v}|} = (1+E_p^D)(1-E_p^C). \tag{3}$$

By a simple calculation we get $E_p^C = \frac{E_p^D}{2+E_p^D}$. So we have proved

Theorem 5. *For every $p \geq 1$ we have*

$$E_p^C = \frac{E_p^D}{2 + E_p^D} = \frac{\sqrt{2p^2 + 2 - 2p\sqrt{p^2+1}} - 1}{\sqrt{2p^2 + 2 - 2p\sqrt{p^2+1}} + 1} = \frac{1}{16p^2} + O\left(\frac{1}{p^4}\right).$$

Remark 2. Observe that E_p^B is about 37% larger than E_p^C. This is the price to be paid for the restriction $W(n,0) = |n|$ for $n \in \mathbb{Z}$. The value of E_p^D is about twice the error E_p^C. This is due to the fact that the negative and positive deviations in E_p^C are added to the positive deviation in E_p^D.

5 Conclusion

In this paper, we have determined the smallest possible maximum relative error of chamfer distances with respect to the Euclidean distance under various conditions. We have dealt with approximating distances from three main aspects: supposing that a horizontal/vertical step has a weight 1 in the local chamfer neighborhoods, majorating the Euclidean distance, and also without any constraint. We have calculated optimal weights for small (5×5 and 7×7) neighborhoods in a certain case, as well. Our framework is embedded in the theory of neighborhood sequences with possible generalizations in this field.

References

[1] Bailey, D.G. An efficient euclidean distance transform. *Lecture Notes in Computer Science*, 3322:394–408, 2004.

[2] Borgefors, G. Distance transformations in arbitrary dimensions. *Computer Vision, Graphics, and Image Processing*, 27:321–345, 1984.

[3] Borgefors, G. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34:344–371, 1986.

[4] Borgefors, G. Hierarchical chamfer matching: a parametric edge matching algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:849–865, 1988.

[5] Butt, M.A. and Maragos, P. Optimum design of chamfer distance transforms. *IEEE Transactions on Image Processing*, 7:1477–1484, 1998.

[6] Coquin, D. and Bolon, Ph. Discrete distance operator on rectangular grids. *Pattern Recognition Letters*, 16:911–923, 1995.

[7] Cuisenaire, O. *Distance Transformation, Fast Algorithms and Applications to Medical Image Processing.* PhD thesis, Université Catholique de Louvain, 1999.

[8] Danielsson, P.E. Euclidean distance mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[9] Das, P.P., Chakrabarti, P.P., and Chatterji, B.N. Distance functions in digital geometry. *Information Sciences*, 42:113–136, 1987.

[10] Fazekas, A., Hajdu, A., and Hajdu, L. Lattice of generalized neighbourhood sequences in nd and ∞d. *Publ. Math. Debrecen*, 60:405–427, 2002.

[11] Fouard, C., Strand, R., and Borgefors, G. Weighted distance transforms generalized to modules and their computation on point lattices. *Pattern Recognition*, 40:2453–2474, 2007.

[12] Hajdu, A., Hajdu, L., and Tijdeman, R. General neighborhood sequences in $\mathbb{Z}^n$. *Discrete Appl. Math.*, 155:2507–2522, 2007.

[13] Jr, C.R. Maurer, Qi, R., and Raghavan, V. A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:265–270, 2003.

[14] Nagy, B. Distance with generalized neighbourhood sequences in nd and ∞d. *Discrete Appl. Math.*, 156:2344–2351, 2008.

[15] Nagy, B. and Strand, R. Neighborhood sequences in the diamond grid: algorithms with two and three neighbors. *International Journal of Imaging Systems and Technology*, 19:146–157, 2009.

[16] Nagy, B. and Strand, R. Neighborhood sequences on nd hexagonal/face-centered-cubic grids. *Lecture Notes in Computer Science*, 5852:96–108, 2009.

[17] Scholtus, S. Chamfer Distances with Integer Neighborhoods, Master Thesis, Leiden University, The Netherlands. 2006.

[18] Scholtus, S. and Tijdeman, R. Chamfer distances with integer neighborhoods. Technical report, Leiden University, The Netherlands, 2006.

[19] Shih, F.Y. and Wu, Y.T. Fast euclidean distance transformation in two scans using a 3×3 neighborhood. *Computer Vision and Image Understanding*, 93:195–205, 2004.

[20] Strand, R., Nagy, B., Fouard, C., and Borgefors, G. Generating distance maps with neighbourhood sequences. *Lecture Notes in Computer Science*, 4245:295–307, 2006.

[21] Thiel, E. *Les Distances de Chamfrein en Analyse d'Images: Fondements et Applications.* PhD thesis, Université Joseph Fourier de Grenoble, 1994.

[22] Verwer, B.H.J. *Distance Transforms: Matrics, Algorithms and Applications.* PhD thesis, Techn. Univ. Delft, 1991.

[23] Verwer, B.H.J. Local distances for distance transforms in two and three dimensions. *Pattern Recognition Letters*, 12:671–682, 1991.

[24] Yamashita, M. and Ibaraki, T. Distances defined by neighbourhood sequences. *Pattern Recognition*, 19:237–246, 1986.

Received 16th February 2010

Acta Cybernetica **20** (2012) 419–440.

Person Attribute Extraction from the Textual Parts of Web Pages

István Nagy T.*

Abstract

We present a web mining system that clusters persons sharing the same name and also extracts bibliographical information about them. The input of our system is the result of web search engine queries in English or in Hungarian. For system evaluation in English, our system (RGAI) participated in the third Web People Search Task challenge [1]. The chief characteristics of our approach compared to the others are that we focus on the raw textual parts of the web pages instead of the structured parts, we group similar attribute classes together and we explicitly handle their interdependencies. The RGAI system achieved top results on the person attribute extraction subtask, and average results on the person clustering subtask. Following the shared task annotation principles, we also manually constructed a Hungarian person disambiguation corpus and adapted our system from English to Hungarian. We present experimental results on this as well.

Keywords: natural language processing, information extraction, web content mining, person attribute extraction, document clustering

1 Introduction

Personal names are among the most frequently searched items in web search engines. At the same time, these types of search results ignore the fact that a name may be associated with more than one person. Sometimes person names are highly ambiguous (see Figure 1).

The first Web People Search challenge (WePS) [2] organized in 2007 focused on this disambiguation problem. As input, the participating systems got web pages retrieved from a web search engine using a given person name as a query. The aim of the task was to find all the different people among the results and assign a corresponding document to each person. The second WePS challenge [3] organized in 2009 contained a new subtask. The attribute extraction subtask [30] was to identify 16 different bibliographical attributes from personal web pages such as the

*University of Szeged, Department of Informatics, 6720 Szeged, Árpád tér 2., Hungary E-mail: nistvan@inf.u-szeged.hu

birth date, affiliation, and occupation. This subtask proved very difficult and the best system only achieved an F-score of 12.2.

The third WePS shared task [1] introduced a novel subtask which sought to mine attributes for persons, i.e. rather than recognizing attributes in web pages, the task was to assign them to people (the clusters of pages belonging to each given person). This attribute extraction task is more difficult than the previous one because aggregation of recognised text spans has to be carried out as well (people may use synonyms, spelling variants etc. when writing the same content on different pages).

Figure 1: Personal Name Disambiguation Problem.

Our system handles the web page clustering and person-level attribute extraction tasks together. Here, we introduce a novel approach that is primarily based on biographical attribute extraction and it uses this information to determine the clusters of persons. Our system participated in the third WePS challenge and achieved top results on the person attribute extraction subtask, and average results on the person clustering subtask. In addition, we present the first Hungarian Name Disambiguation Corpus, which is based on the WePS3 challenge corpora. We found that the results on the Hungarian corpus are comparable to the English ones.

In Section 2 we present some related work, which is followed by a description of our methods in Section 3. Experimental results are shown in Section 4 and discussed in Section 5, then in Section 6 we draw some conclusions and make some suggestions for future study.

2 Related work

Text mining and information extraction approaches can quite effectively solve the problem of automatic attribute extraction from machine-readable documents. Text mining [12] involves the extraction of hidden or non-trivial information from unstructured (or partially structured) electronic text files. It means the automatic extraction of latent and hidden correlations or knowledge from available data sets. Text mining problems are special and require different solutions. Modeling the structure of natural language-written texts syntactically and semantically is essential and an examination of the styles of these texts is also required.

Among the text mining applications, Web-mining [21] makes use of the internet, which is the world's largest and fastest-growing repository. There are four knowledge discovery domains that pertain to web mining. There are Web Content Mining [13, 24], Web Structure Mining [20, 6], Wrapper Induction [37, 22] and Web Usage Mining [26, 31]. The aim of Web Content Mining is to extract useful information from the natural language-written parts of websites. Web Structure Mining is the process of using graph theory to analyze the node and connection structure of a website. The aim of Wrapper Induction is to automatically extract data from structured data (HTML, DOM). It is based on automatically detecting the structure (e.g. webshops, product information pages or similar documents), and extracting relevant information. However, Web Usage Mining focuses on techniques that could predict user behavior while the user interacts with the web. The approaches introduced in this paper fall into the category of Web Content Mining.

The initial classical Web Content Mining attempts appeared around 1998-'99 [13, 24]. These were basically rule-based systems either hand-crafted rules, or supervised learning rules trained on a manually annotated corpus. The next-generation approaches were weakly supervised learning methods. Here, the input is a seed list of target information pairs and the goal is to gather pairs which are related to each other in the same way as seed pairs. These pairs may contain related entities like country - capital city [15] and celebrity partnerships [9] or form an entity-attribute pair like Nobel Prize recipient - year [16] or may be concerned with retrieving all available attributes for entities [5, 28]. These systems are usually downloaded from the websites which include the current pairs, and syntactic/semantic rules are learned from their sentences. Next, a web corpus is used in the pre-learned patterns to obtain a new pair. These seed-based systems exploit the redundancy of the WWW. They are based on the hypothesis that useful information is available in several places and in several forms on the Internet, hence some very accurate rules for boosting information extraction efficiency were required. Their goal is to find and recognize at least one occurrence of the target item and not to find every occurrence on the web. But it is not the case in biographical attribute extraction from related person's pages. Here, we must capture each mention of the item of information.

A good example of a web-mining application is the portal for job seekers developed by FlipDog.com. That is, job vacancies are automatically collected from 60,000 company websites. Furthermore, they publish an analysis each month about

their database trends and changes. The services of this website are used by many organizations because similar comprehensive and timely reviews are unavailable elsewhere. The automatically extracted information about jobs proved to be sufficient to achieve good accuracy, while data from less reliable web pages can be checked manually [25].

One task for text-mining systems is identifying Named Entities (person, organization, location names) in documents, as they usually play an important role. Named Entity recognition was a task in the framework of the Message Understanding Conference MUC-7 [10]. Participants had to identify personal names, geographical names, organizations, and other names categorization and time, quantity, and descriptive terms. In 2003, The Conference on Computational Natural Language Learning (CoNNL) [29] was announced by the open tournaments. The aim was the construction of a Named Entity Recognition (NER) model which could handle English and German texts. One of the most successful and most widely used approaches for sequence labeling is the Conditional Random Fields (CRF) approach [23]. For the Hungarian language, there are existing rule-based [11] and machine learning Named Entity Recognition tools [32, 35]. The statistical systems are based on the Hungarian Named Entity Corpus of Business Newswire Texts [32]. Our system contains Conditional Random Fields-based NER modules that are required for information processing and extraction.

For the person clustering subtasks, the majority of the shared task participants used some pre-processing step before the individual documents could be represented, then some general clustering algorithm was applied [3]. One of the main problems with the person disambiguation problem is deciding how to determine the number of entities present. The number of clusters was estimated by different approaches during the WePS challenges such as cross-validation, permutation tests, resampling and panelized likelihood [8, 14]. However, these methods only work well for clusters that are well separated and thus they all proved inefficient. Some participant systems used hierarchical clustering approaches [17, 7], so they did not have to estimate the number of clusters.

Rule-based approaches [7] achieved the best results for the attribute extraction task of WePS2. However, the two-step methods [36, 19] also did well. These approaches first mark the potential values of attributes, and then in the second step, validators have to decide about the attributes. Other participants used manual and automatic pattern constructions [4].

3 Our method

We shall focus on the raw text parts of the web pages because we found empirically that more pages express content in textual form than in structured form. The first step of information extraction may be to construct a good section selection module. When handling the problem, we first extract the candidate attributes from the relevant sections of web pages, then we cluster the pages by merging clusters having common person attributes and aggregate attributes with the persons identified.

3.1 Preprocessing

The input of each participant's system was a set of pages retrieved from a web search engine using a given person's name as a query. We assumed that useful information was available in the natural language-written part of websites and tables [27]. This is why we concentrated on the natural language-written part of websites and tables, and we discarded a lot of noisy and misleading elements from pages (e.g. menu elements). These elements can seriously hinder the proper functioning of Natural Language Processing (NLP) tools.

In order to identify textual paragraphs, we applied the Stanford POS tagger [34] to each section of the DOM tree of the HTML files. We assumed that one piece of text was a textual paragraph if it was longer than 60 characters and it contained more than one verb. We extracted several attributes with our own Named Entity Recognition [33] system which was trained on CoNLL-2003 training data sets. When we used this model on the entire set of paragraphs, the accuracy score obtained was low. To handle this problem we developed attribute-specific, relevant section selection modules. Then we looked for the occurrences of all gold standard attributes using simple string matching in each extracted paragraph. In this way, we created a database with positive and negative paragraphs for the actual attribute. Then we created a set of positive words with the most frequently occurring words taken from the positive paragraphs. If a paragraph from the prediction document set contained at least one word from the actual positive list, we marked it as a positive paragraph and we only extracted attributes from these paragraphs. This approach was used to find the occupation, affiliation, award and school attributes.

Table 1: Definition of attributes of Person for the WePS attribute extraction task

Attribute Class	**Examples of Attribute Value**
Date of birth	4 February 1888
Birth place	Brooklyn, Massachusetts
Other name	JFK
Occupation	Politician
Affiliation	University of California, Los Angeles
Award	Pulitzer Prize
School	Stanford University
Major	Mathematics
Degree	Ph.D.
Mentor	Tony Visconti
Nationality	American
Relatives	Jacqueline Bouvier
Phone	+1 (111) 111-1111
Fax	(111) 111-1111
Email	xxx@yyy.com
Web site	http://rgai.inf.u-szeged.hu/

3.2 Attribute extraction

This subtask involves extracting 16 kinds of "attribute values" for target individuals whose names appear on each of the web pages provided. The attribute types are listed in Table 1.

Our attribute extraction system consists of two main parts, namely a candidate attributes extraction module and an attribute verification module. Based on this approach, we first mark potential attribute values in a paragraph. Then we find out which candidate values have been found.

When handling this subtask of attribute extraction, it seems necessary to classify the attribute classes. Hence, we aggregated similar attributes into logical groups. For instance, the name group contains the *other name*, *relatives* and *mentor* attribute classes. One advantage of this typology scheme is that we can apply the same approach when we extract the same type of attributes. Another is that we can assume subordinate relations among the coherent attributes. For example, we only marked a candidate name as *mentor* if it was not *relatives* or *other name*.

Table 2: Attribute typologies

Name	Availability	Organization
other name	web page	award
mentor	phone number	affiliation
	fax	

Next, we will elaborate on the extraction procedure for each of the attributes.

Date of birth: If a paragraph contains *born*, *birth* or *birthday* phrases, we find candidate dates with a date validator within a window of the word. This validator works with 9 different regular expression rules, and can identify dates written in different formats in the span of text.

Birth place: When a paragraph contains *born*, *birth*, *birthplace*, *hometown* and *native* phrases, we use the location markups given by the NER tool [33] trained on the locations class of the CoNLL-2003 training data set to identify candidate locations for the birthplace. We accept a location as a birthplace if a birthplace validator validates it.

Occupation: According to the WePS2 results, it was one of the most difficult, ambiguous and frequent attribute classes, which is due to the abstract nature of this attribute. Hence we avoided using lists. It was not available in any NER model or training database. So we created a training database by matching all gold annotations to paragraphs. We used simple string matching and we did not know where the actual attribute occurred. However, the resulting data set was very noisy. We trained our NER tool [33] on this training database, and we applied it on the candidate occupation paragraphs.

Organizations (*school*, *award*, *affiliation*): We found that these types of attributes were names of organizations so we grouped them together. We also used an NER

tool [33] here trained on the organization class of the CoNLL-2003 training data to identify candidate organization mentions only in affiliation-candidate paragraphs. When the NER model marks a candidate organization phrase, we first search for the school attribute. Then a potential candidate organization is marked as a school if it appears near some cue phrases such as *graduate*, *degree*, *attend*, *education* and *science*. Next, we defined a school validator that uses the MIVTU [36] school word frequency list with *School*, *High*, *Academy*, *Christian*, *HS*, *Central* and *Senior*. We extended this list with *University*, *College*, *Elementary*, *New*, *State*, *Saint*, *Institute* phrases. First letter capitalized sequences, except for some stopwords like *of* and *at* which contain at least one of these words, were marked as a school by a validator. If the school validator did not validate the potential candidate organization, we looked for the award attribute. When candidate sequences appear near cue phrases such as *award*, *win*, *won*, *receive* and *prize*, we assumed an expression with *award* was an attribute. We also defined an award validator that validates a first letter capitalized sequence except for some stopword like *at* or *of*, if it contains at least one element of the *award*, *prize*, *medal*, *order*, *year*, *player* and *best* phrases. When the candidate string is not a valid *school* and *award*, we tag it to the *affiliation* attribute.

Degree: A list of degrees complied manually which contains 62 items. When we found one element from these lists in a paragraph, we marked it as a degree attribute. We assumed that the degree attribute might be located far from the name in a CV-type web page.

Names (*relatives*, *other name*, *mentor*): These types of attributes are person names so we found that they occur together. To identify name attributes we used an NER tool [33] trained on the person names of the CoNLL training data. A model extracts name phrases as relatives if they appear in the immediate context of the candidate that indicates various relationships like *father*, *son*, *daughter* and so on. Cue phrases were the same as in the MIVTU [36] system used in WePS2 and are also available in Wikipedia. Sometimes we did not mark the potential candidate sequence for *relatives*, but looked for *other name* attributes instead. We hypothesized that a person does not write his or her name using the same number of tokens; at the same time *other name* has to contain at least a part of the original name. For instance when the original name was *Helen Thomas*, we did not accept the candidate string *Helen McCumber*, but we accepted the *Helen M. Thomas* sequence. This hypothesis may not be true for nicknames. If a name was not marked as *relatives* or *other name*, we analyzed the potential candidates for a *mentor* name. If it appeared near cue phrases such as *study with*, *work with*, *coach*, *train*, *advis*, *mentor*, *supervisor*, *principal*, *manager* and *promote* we marked the potential candidate sequence as a mentor attribute.

Nationality: We created a list of nationalities that contained 371 elements. It has multiple entries for certain nationalities. Once we had found one element from this list in a paragraph or table, we assumed it was a potential nationality attribute. Then we selected the most frequent potential nationality attribute of the web pages. When extracting *availability* attribute classes we did not focus just on textual paragraphs, but examined the whole text of web pages as these types of attributes may

occur in other parts as well.
Phone: When a text contains *tel*, *telephone*, *ph:*, *phone*, *mobile*, *call*, *reached at*, *office*, *cell* or *contact* words or a part of the original name, we applied the following regular expression:

```
(((?[0-9+(][.()0-9s/-]4,[0-9])((?(s?x|s?ext|s?hart).?)?  d1,5)?)
```

It is a permitted regular expression for potential phone numbers. We defined a phone number validator that validated the sequence determined by the regular expression.
Fax: We use the same method as that for phone numbers, except for that we look for *fax*, *telfax* and *telefax* phrases.
E-mail: We assumed that if somebody offers their e-mail address, it is also a link. Therefore, we examined links that contained the mailto tag. Moreover, we assumed that every mail address contains the original name or one part of the original name. Hence we defined an e-mail address validator that validates email addresses. We generate all character trigrams from the original name and when an e-mail address contains at least one of them, the validator accepts it. We defined a stopword list as well. This list contains words such as *wiki*, *support* and *webmaster*. Should a candidate e-mail address contain one from the stopword list, the validator does not accept it. Next we extracted the domain from all accepted e-mail addresses, which we used for the website attribute.
Website: We assumed that when somebody displays a web address on a website, it is also a link, so a web address is a link at the same time. In this case we only extract a website attribute from links. We marked a potential candidate attribute as a website when it contained the original name or one part of the original extracted domain name from the e-mail attribute.

3.3 Person disambiguation

Our chief hypothesis in the person disambiguation subtask was that it can be effectively solved by using extracted person attributes. We defined a weighting of attribute classes. The most useful attribute classes were *web address*, *e-mail*, *telephone*, *fax number* and *other name* and they got a weight of 3. In addition, we weighted *birth date* as 2, while *birth place*, *mentor*, *affiliation*, *occupation*, *nationality*, *relatives*, *school*, and *award* each got a weight of 1. Then every document was represented by a vector with extracted person attribute values.

To define a document similarity measure, we needed normalize the attribute to values. We developed individual normalization rules for each attribute class. For example, the birth place of *United States of America* could be referred to as *USA*, *U.S.A.*, *United States*, *Federal United States* and so on. We created a synonym dictionary based on the re-direct links of the English Wikipedia. We developed regular expressions or rules based on normalization procedures for other attribute classes.

As a first approach for web page clustering, a bottom-up heuristic clustering was performed based on this similarity measure. Here, the starting clusters consist of the individual web pages and then the clusters are merged iteratively until a stopping criterion is reached. For each step of this procedure the most similar clusters are merged (the union of their attributes formed the attribute set of the resulting cluster), where the similarity measure of the weighted size of the intersection of the cluster attribute sets was employed. The stopping criterion was defined to be a similarity value threshold of 2, i.e. if the similarity value of the closest clusters is less than 2, the procedure is terminated (`RGAI(A+H)` with attributes as features and hierarchial clustering).

Besides this heuristic bottom-up clustering, we employed the Xmeans algorithm from the WEKA Java package [18] as well. The advantage of this approach is that it is not necessary to define the number of clusters, but we can define the minimum number of clusters. We used the final number of clusters obtained by heuristic clustering as the minimum number of clusters for Xmeans with (`RGAI(A+X)` attributes as features and Xmeans).

We compared our person attribute approach against standard document clustering methods. With the results of `RGAI(S+X)` (snippet as features), we only used the search engine snippet data. These types of representation compress the most important pieces of information. We represented the data with the tf-idf vector space model, where each document is represented by a vector of its token uni- and bigrams and employed Xmeans for clustering. Next, `RGAI(AS+X)` is a hybrid method of the above two approaches, i.e. the feature sets of the person-based attribute and the snippet-based clustering were merged.

We evaluated our attribute-based approach presented above on the Hungarian Name Disambiguation Corpus as well. To compare our method against the document clustering approach, we employed the KMeans algorithm in the WEKA Java package, as we have more information about the test corpus in this case. This algorithm, like other clustering algorithms, requires fixing the number of clusters beforehand. Since this value is not known in this task, we used various heuristics to estimate it. First (`HNum`), we used the final cluster number of the bottom-up clustering output as the number of clusters for KMeans. In the second case (`Simple`), the average number of documents associated with one name was employed (7 on the corpus). After, we applied the gold standard number of clusters (as determined by human annotators) related to each name in the corpus (`Oracle`).

Lastly, we used the two simple clustering baseline methods, which are shown in Figure 2. In the first case, all the documents are assigned to a single cluster and in the second case, each document is assigned to a different cluster.

3.4 Attribute aggregation

We had to aggregate those attributes that occurred in web pages and were found in a cluster, i.e. belonged to a particular person. The official evaluation metric of the challenge required only one attribute from each class. As we extracted more than one potential attribute value for each class, we had to choose one (e.g. a person

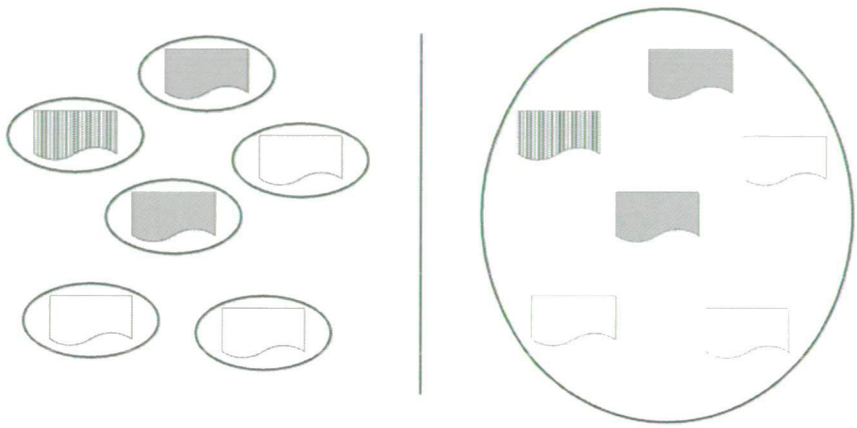

Figure 2: Baseline methods.

may mention several of his affiliations). In the end we chose the most frequent element per person from each attribute class. When some attribute frequencies were the same, we just chose one at random.

4 Results

In this section we describe the results achieved on the Hungarian Name Disambiguation Corpus and the third WePS campaign test corpus.

4.1 Evaluation issues

In the first WePS, the standard clustering metrics, Purity and Inverse Purity [2] was used to evaluate participant systems against a manually annotated test corpus. During this first evaluation campaign, the organizers concluded that it was possible to cheat using these metrics. The problem is caused by the overlap of the clusters. If a page referred to more than one person, the organizers were allowed to include a document in several clusters. Exploiting this issue, higher scores could be obtained by simply forming one cluster for each document along with one cluster containing all the documents. In this way, a maximal value of Inverse Purity could be obtained together with a high score of Purity. Figure 3 shows an example of the difference between the cheating approach and a correct approach. Since each document is a separate cluster, the cheating system will achieve a maximum value for Inverse Purity. However, it may result in a low Purity score. Since a page can refer to more than one person, it can specify a maximum value of the Purity cluster (for each item). In this way, the average Purity value can be higher.

To overcome of these shortcomings, the organisers of the campaign chose the extended B-Cubed metric for the clustering evaluation metric. We employed this B-cubed metric to evaluate the clustering results of the Hungarian dataset. In this case, we also investigated the precision and recall. However, it is necessary to extend the correctness when a document is classified in several clusters. This is why we defined the multiple version for accuracy and coverage by:

$$\textit{Multiple recall(e, e')} = \frac{Min(|C(e) \cap C(e')|, |L(e) \cap L(e')|)}{|L(e) \cap L(e')|}$$

$$\textit{Multiple precision(e, e')} = \frac{Min(|C(e) \cap C(e')|, |L(e) \cap L(e')|)}{|C(e) \cap C(e')|}$$

where e and e' are two different elements, $L(e)$ is the set of categories and $C(e)$ is the set of clusters associated with e. Multiple precision can only be used if e and e' share some cluster, and multiple recall can be used when e and e' share some category. The previous value is maximal (1) when the number of shared categories is lower than or equal to the number of shared clusters, and it is minimal (0) when the two items do not share any category. Here, the multiple recall is maximal when the number of shared clusters is lower or equal to the number of shared categories, and it is minimal when the two items do not share any cluster.

For the attribute extraction subtask, the attributes were only extracted from documents that grouped person A or person B in the clustering annotation. Evaluation metrics were computed as follows:

Precision: For a given person, this is the number of correct attribute/value pairs divided by the total number of attribute/value pairs extracted.

Recall: For a given person, this is the number of attributes having at least one correct value divided by the total number of attributes for which a correct value has been found by at least one of the systems.

F-score: 1 / (alpha * 1/prec + (1-alpha) * 1/rec), where alpha was 0.5.

The above-defined given person is taken from the evaluation for the clustering subtask. Here, every cluster has 2 person names. During the evaluation process for each person the most resampled cluster is defined as the cluster with the higher F-score or with the highest recall, where

Precision: The number of documents in the cluster that refer to the person / number of documents in cluster.

Recall: The number of documents in the cluster that refer to the person / number of documents that refer to the person.

The organizers found that these methods often missed the cluster with more relevant attributes, resulting in extremely low evaluation results. Hence they defined another metric: they choose the cluster with the best recall of attributes for a person. Next, the organizers defined two different interpretations of the manual annotations, which were combined with the other two clustering evaluation options.

Strict evaluation: We count as correct all attribute - value pairs judged as correct by a majority of annotators and as incorrect otherwise.

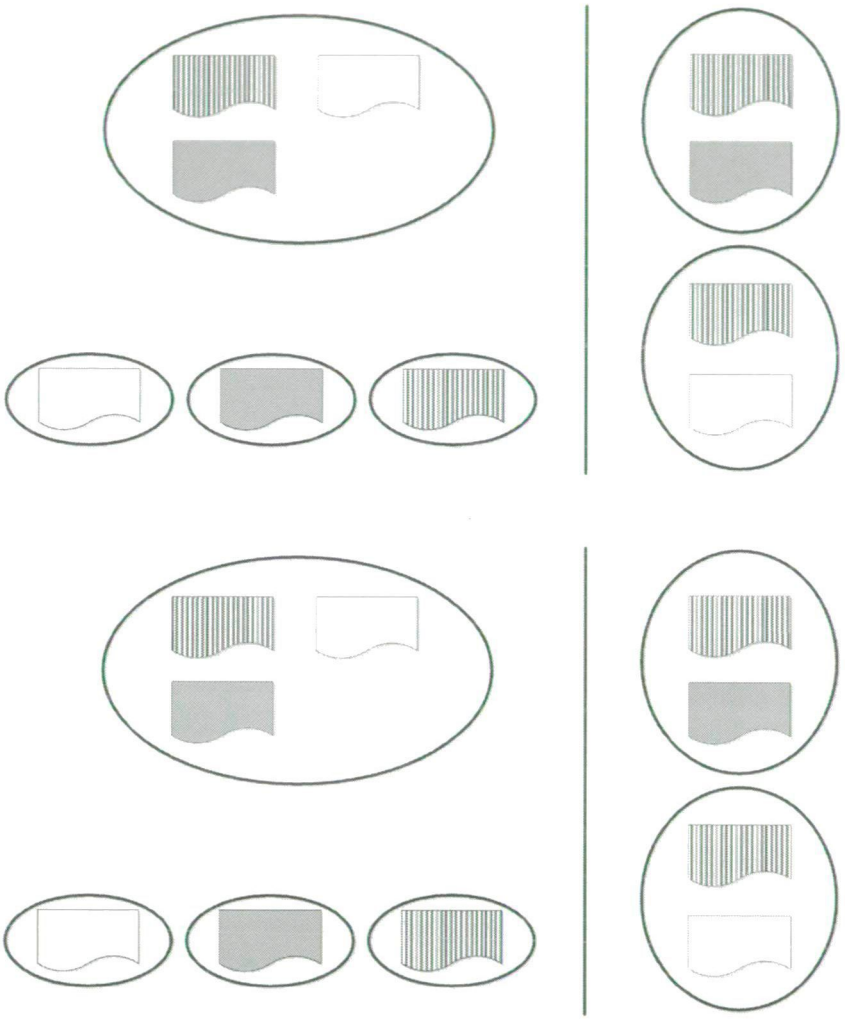

Figure 3: Output of cheat distribution vs. correct solution.

Lenient evaluation: We count as correct all attribute - value pairs judged as correct or inexact by a majority of annotators, and as incorrect otherwise.
On the attribute extraction subtask, the official evaluation metric was the attribute recall based clustering with lenient manual annotation. These results are shown in Table 5, where `RGAI(A+X)` achieved the top F-score.

4.2 The Hungarian Test Corpus

To evaluate our system, we created the Hungarian Name Disambiguation Corpus with manually annotated web pages. It consists of 10 different Hungarian names and it is available [1] at `http://www.inf.u-szeged.hu/rgai/disambiguation`.

The name list included several well-known figures like *Sándor Csányi* (president of the largest bank in Hungary and a famous Hungarian actor) and the most common names in Hungary like *István Kovács* and *Zsófia Szabó*. When we selected common names, we looked for names of famous people. As in the first case, it could be a famous boxer or in the second case a popular actress. The list included Hungarian President *Pál Schmitt*, a former Olympic champion, who has held several top governmental positions over the years. Tracking down *Pál Schmitt*-related websites can be a hard task because websites write about him in different areas of life.

We downloaded 100 web pages related to each name using the Yahoo search engine[2]. During the annotation process, the human annotators could assign 570 pages to a specific person. So an average of 57 pages was related to each name. However, there were big differences for the various names. For example, in the case of *Miklós Zrínyi*, a historical Hungarian person, the results are mostly related to an institute bearing his name or in the case *Pál Schmitt* the pages focused on just only one person. In total, 120 different individuals were identified by the annotators. In this case there were big differences again for the various names. Over 30 different individuals had the common Hungarian name *Istvács Kovács*, while in the *Pál Schmitt* case the results were mostly related to one person.

4.3 Results on the Hungarian Test Corpus

The results achieved by different cluster number initializations are shown in Table 3. As can be seen, our attribute-based document clustering approach achieved the best results on the Hungarian Name Disambiguation Corpus. When we got the correct number of clusters (*Oracle*), it achieved the best results among the clustering methods. The other two approaches yielded the same F-score with different precision and recall values.

4.4 The English Test Corpus

The WePS3 datasets containing 300 names were used for testing, in contrast to WePS2 where the test database consisted only of 30 names. The person names were obtained from a US Census, Wikipedia, Computer Science PC lists and names which contained at least one person who is a lawyer, corporate executive or estate agent. For each name the top 200 web search results from the Yahoo! API were downloaded and archived with their corresponding search metadata, like search snippet, title, URL and position in the results ranking.

[1] under CC license

[2] www.yahoo.com

Table 3: Person disambiguation results on the Hungarian corpus

	average B-Cubed		
Approach	**precision**	**recall**	**F-score**
A+H	**0.59**	**0.64**	**0.59**
All_In_One	0.43	0.84	0.50
Oracle	0.59	0.37	0.43
Simple	0.52	0.38	0.36
HNum	0.69	0.28	0.36
One_in_One	0.93	0.24	0.35

During the annotation process, only two-person-related websites were labeled by the annotators. In this way, the annotation effort was radically reduced. Consequently, large amounts of human resources and time were saved. Clearly, the gold standard used was not perfect. When the Hungarian Name Disambiguation Corpus was created we avoided this problem and the annotators had to identify each individual person.

The attribute extraction subtasks were not manually annotated in the gold standard test database of WePS3. They were only manually evaluated for the participant systems' outputs. Only the clustering subtask of finding person-related attributes was evaluated. Then an annotator got a website with the ten most common attributes - value pairs according to the participant systems, and he or she had to decide which attribute belonged to which of the following categories:

- Correct: the attribute appears in the website and it is related to the actual person.
- Incorrect for any reason other than being too long or too short. For instance, the type of attribute is incorrect (e.g. a number is incorrectly identified as a telephone number); the attribute is not related to the actual person (e.g. the attribute describes some other person described on the page); or the attribute simply did not appear in the text.
- The attribute is correct, but it is too long or too short. So the attribute has one of the following problems:
  - It is too short. The attribute is incomplete (e.g. *director* when it should say *director of marketing*).
  - It is too long. The attribute contains a correct value but includes irrelevant information (e.g. *CEO in 1982* when it should say just *CEO*).
- Cannot decide, because the web page is unreadable for some reason.
- The web page is readable, but the specified person is not on this page.

4.5 Results on the English Test Corpus

Here, we describe the results obtained on the WePS3 challenge.

Clustering results on the English Test Corpus

During the evaluation of the clustering subtask the organizers used the extended versions of BCubed Precision and Recall, which was the official evaluation metric with alpha set to 0.5. They evaluated the clustering of documents for each query, just focusing on two different people, except for 50 names, where only documents about one person were considered. The official results on the clustering task of the RGAI systems, other participants and the two baselines results are shown in Table 4. Here, our `RGAI(S+X)` system achieved the best scores.

Table 4: Document clustering results on the English corpus

		average B-Cubed		
rank	**System**	**preceison**	**recall**	**F-score**
1	YHBJ_unofficial	0.61	0.60	0.55
2	AXIS	0.69	0.46	0.50
3	TALP	0.40	0.66	0.44
4	**RGAI(S+X)**	**0.38**	**0.61**	**0.40**
5	WOLVES	0.31	0.80	0.40
6	RGAI(S+X)	0.38	0.61	0.40
7	RGAI(A+H)	0.40	0.57	0.40
8	DEADELUS	0.29	0.84	0.39
9	BYU	0.52	0.39	0.38
10	RGAI(A+X)	0.47	0.43	0.38
11	RGAI(AS+X)	0.36	0.55	0.38
	one_in_one_baseline	1.00	0.23	0.35
12	HITSGS	0.26	0.81	0.35
	All_in_one_baseline	0.22	1.00	0.32

Attribute extraction results on the English corpus

The WePS2 train and test sets were used for the training set, which contained 5,122 websites with 187,032 paragraphs. We found 2,781 affiliations, 3,419 occupations and 2,092 biographical paragraphs. For the location, organization and names, markups given by the NER tool [33] trained on the CoNLL-2003 training data set, it achieved F-scores of 89.94% on names, 87.06% on locations and 76.37% on organizations evaluated on the CoNLL-2003 evaluation set [29].

As Table 6 shows, the results of the RGAI systems when the clustering resemblance was the recall approach and the manual annotation was lenient. Our

Table 5: Lenient annotation and attribute recall based clustering

System	precision	recall	F-score
RGAI(A+X)	**21.53**	**24.53**	**18.46**
RGAI(S+X)	17.99	19.46	14.72
Intelius_AE_UNOFFICIAL	16.48	17.18	13.15
RGAI(AS+X)	14.95	16.46	12.52
RGAI(S+X)	16.37	15.09	12.45
RGAI(A+H)	15.46	15.33	12.19
BYU	11.11	13.94	9.86
WOLVES_AE_1	18.49	8.83	9.60

best result was achieved by the `RGAI(A+X)` system, but the Intelius system was outstanding.

Table 6: Lenient annotation with recall based clustering

System	precision	recall	F-score
Intelius_AE_UNOFFICIAL	10.66	14.55	9.93
RGAI(A+X)	**5.81**	**8.99**	**5.93**
WOLVES_AE_2	5.42	5.99	4.55
RGAI(AS+X)	3.99	7.56	4.38
RGAI(S+X)	3.84	7.51	4.37
WOLVES_AE_1	4.96	4.77	4.14
RGAI(A+H)	3.76	5.15	3.67
RGAI(S+X)	3.82	5.27	3.55
BYU	2.86	5.16	3.03

However, when we used the lenient annotation interpretation and the clustering approach based on the F-score, our `RGAI(A+X)` system achieved significantly better results.

When the clustering method was attribute recall based and the annotation was strict, the `RGAI(A+X)` system gave the top F-score.

When we used the strict annotation and recall-based clustering approach, the results of the Intelius system were noticeably better than those of the other systems. It was able to cluster the documents better (see Table 9).

Next, Table 10 shows the results got when the clustering approach was based on the F-score and the annotation was strict. `RGAI(A+X)` achieved the best results, but the other systems performed fairly well.

The above tables indicate that our approach achieved an F-score slightly above 10 on the F-score-based clustering. Compared to the WePS2 results – where the best system achieved about an F-score of twelve – these results are competitive as

Table 7: Lenient annotation with F-score based clustering

System	precision	recall	F-score
RGAI(A+X)	**13.22**	**15.48**	**11.67**
RGAI(S+X)	12.14	13.23	10.06
RGAI(AS+X)	11.26	11.89	9.38
RGAI(S+X)	11.58	10.42	8.71
WOLVES_AE_1	15.17	7.09	8.14
RGAI(A+H)	9.97	9.37	7.57
Intelius_AE_UNOFFICIAL	7.44	7.23	6.39
BYU	5.99	7.88	5.59
WOLVES_AE_2	7.46	6.42	5.35

Table 8: Strict annotation and attribute recall based clustering

System	precision	recall	F-score
RGAI(A+X)	**21.38**	**24.48**	**18.40**
RGAI(S+X)	17.94	19.46	14.69
Intelius_AE_UNOFFICIAL	16.41	17.19	13.13
RGAI(AS+X)	14.88	16.41	12.47
RGAI(S+X)	16.37	15.11	12.47
RGAI(A+H)	15.43	15.34	12.20
BYU	11.08	13.98	9.86
WOLVES_AE_1	18.60	8.83	9.62
WOLVES_AE_2	7.74	7.14	5.79

Table 9: Strict anotation with recall based clustering

System	precision	recall	F-score
Intelius_AE_UNOFFICIAL	9.81	14.37	9.60
RGAI(A+X)	**5.81**	**8.83**	**5.89**
RGAI(S+X)	4.20	7.98	4.76
WOLVES_AE_2	5.40	5.99	4.54
RGAI(AS+X)	3.97	7.53	4.36
WOLVES_AE_1	4.94	4.75	4.12
RGAI(A+H)	3.76	5.16	3.67
RGAI(S+X)	3.82	5.28	3.56
BYU	2.81	5.14	2.99

Table 10: Strict annotation with F-score based clustering

System	precision	recall	F-score
RGAI(A+X)	**13.00**	**15.80**	**11.72**
RGAI(S+X)	12.25	13.28	10.13
RGAI(AS+X)	10.90	11.35	8.98
RGAI(S+X)	11.48	10.52	8.79
WOLVES_AE_1	15.61	7.14	8.20
RGAI(A+H)	9.97	9.64	7.80
Intelius_AE_UNOFFICIAL	7.28	7.10	6.10
BYU	6.05	8.11	5.68
WOLVES_AE_2	7.44	6.42	5.34

we solved a more complex problem here. Nevertheless, the recall-based results tell us that our clustering approach needs to be improved.

5 Discussion

On the Web People Search Clustering Task our system managed to achieve a B-Cubed F score of 40, which is largely due to the annotation process of the test corpus. Although the participants' systems were designed to create the best possible clustering of the whole corpus, during the annotation process only two of the websites related to each person were manually labeled by the annotators. In this way, the organizers spared a large amount of human resources and time, but the gold standard was not perfect. When the Hungarian Name Disambiguation Corpus was created, it did not address this problem and the annotators had to identify each website belonging to the given person. The annotation process did not require large resources because it was a much smaller corpus than the WePS gold standard. Our system achieved a B-Cubed F score of 59 on this corpus, so the development of the gold standard had a beneficial effect on the results. However, a comparison of our results on corpora with different languages suggests that our method is roughly language independent.

The attribute extraction task proved very difficult as the best system only achieved an F-score of 12.2 on the second WePS. The third WePS shared task introduced a novel, harder subtask which sought to mine attributes for a person. In this case it was a major challenge to decide which extracted attribute was related to the actual person and which attribute was related to someone else. We found that for some of the attribute classes (like *occupation*), rule-based approaches outperform machine learning approaches in extracting for the given attribute because these attribute classes are too general. However, the rule-based methods were not perfect, and they are not language independent. We think that an adequate accuracy can be attained just by employing procedures which incorporate external

semantic knowledge.

We were able to extract some attributes better (like *mentor*, *other name*, or *relatives*) via machine learning approaches when we placed the attribute classes into different logical groups and we could assume subordinate relations among the coherent attributes.

6 Summary and Conclusions

In this article, we presented a joint approach for the person name disambiguation and attribute extraction tasks. Our method is based on the useful biographical attributes obtained from the natural language-written parts of websites. We defined 16 different attribute classes, which were extracted by automatic tools. Our method yielded a B-Cubed F-score of 59 on the first Hungarian Name Disambiguation Corpus, while on the WePS3 clustering task it achieved a B-Cubed F-score of 40. These differences can be partly explained by the annotation differences. For the attribute extraction task, our method efficiently extracted the different types of attributes from web pages and we achieved top results on the WePS3 challenge. We think that the two main reasons for the success of our attribute extractor are the following. First, our approach groups attribute classes and introduces rules which efficiently handle the interdependencies among these classes. Second, we focused on the textual parts of the web pages using NLP tools, which demonstrates that raw text parts of person web pages should be analyzed along with the structured parts of the pages.

Although our results seem satisfactory, there are several possible directions for further improvements of the system. One is that the system could be extended with useful biographical information from the table parts of websites. Another is that the validators could be extended by performing a deeper syntactic analysis. Yet another is that an identification of the subjects of the paragraphs seems necessary. Some of the errors that arose were due to the hypothesis that all extracted attributes are related to the actual person, which is not always the case, hence it should be reconsidered.

7 Acknowledgments

This work was supported by the project BELAMI financed by the National Innovation Office of the Hungarian government. The author would like to thank Richárd Farkas for his useful comments and suggestions on a draft version of this paper.

References

[1] Artiles, Javier, Borthwick, Andrew, Gonzalo, Julio, Sekine, Satoshi, and Amigó, Enrique. WePS-3 Evaluation Campaign: Overview of the Web People

Search Clustering and Attribute Extraction Tasks. In *Conference on Multilingual and Multimodal Information Access Evaluation (CLEF)*, 2010.

[2] Artiles, Javier, Gonzalo, Julio, and Sekine, Satoshi. The SemEval-2007 WePS Evaluation: Establishing a benchmark for the Web People Search Task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. ACL, 2007.

[3] Artiles, Javier, Gonzalo, Julio, and Sekine, Satoshi. WePS 2 Evaluation Campaign: overview of the Web People Search Clustering Task. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

[4] Balog, Krisztian, He, Jiyin, Hofmann, Katja, Jijkoun, Valentin, Monz, Christof, Tsagkias, Manos, Weerkamp, Wouter, and de Rijke, Maarten. The University of Amsterdam at WePS2. In *Second Web People Search Evaluation Workshop (WEPS 2009)*, 2009.

[5] Bellare, Kedar, Talukdar, Partha Pratim, Kumaran, Giridhar, Pereira, Fernando, Liberman, Mark, McCallum, Andrew, and Dredze, Mark. Lightly Supervised Attribute Extraction for Web Search. In *Proceedings of Machine Learning for Web Search Workshop, NIPS 2007*.

[6] Chakrabarti, Soumen, Dom, Byron E., Gibson, David, Kleinberg, Jon, Kumar, Ravi, Raghavan, Prabhakar, Rajagopalan, Sridhar, and Tomkins, Andrew. Mining the Link Structure of the World Wide Web. *IEEE Computer*, 32:60–67, 1999.

[7] Chen, Ying, Lee, Sophia Yat Mei, and Huang, Chu-Ren. PolyUHK: A Robust Information Extraction System for Web Personal Names. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

[8] Chen, Ying and Martin, James H. CU-COMSEM: Exploring Rich Features for Unsupervised Web Personal Name Disambiguation. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 125–128, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[9] Cheng, Xiwen, Adolphs, Peter, Xu, Feiyu, Uszkoreit, Hans, and Li, Hong. Gossip Galore - A Self-Learning Agent for Exchanging Pop Trivia. In *EACL (Demos)*, pages 13–16. The Association for Computer Linguistics, 2009.

[10] Chinchor, Nancy. MUC-7 Named Entity Task Definition. *Proceedings of Seventh Message Understanding Conference*, 1998.

[11] Chiticariu, Laura, Krishnamurthy, Rajasekar, Li, Yunyao, Reiss, Frederick, and Vaithyanathan, Shivakumar. Domain adaptation of rule-based annotators for named-entity recognition tasks. In *Proceedings of EMNLP 2010*, pages 1002–1012, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[12] Cohen, K. Bretonnel and Hunter, Lawrence. Getting Started in Text Mining. *PLoS Comput Biol*, 4(1):e20, 01 2008.

[13] Cooley, R., Srivastava, J., and Mobasher, B. Web mining: Information and pattern discovery on the world wide web, 1997.

[14] Elmacioglu, Ergin, Tan, Yee Fan, Yan, Su, Kan, Min-Yen, and Lee, Dongwon. PSNUS: Web People Name Disambiguation by Simple Clustering with Rich Features. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 268–271, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[15] Etzioni, Oren, Cafarella, Michael, Downey, Doug, Popescu, Ana-Maria, Shaked, Tal, Soderl, Stephen, Weld, Daniel S., and Yates, Er. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. *Artificial Intelligence*, 165:91–134, 2005.

[16] Feiyu Xu and Hans Uszkoreit and Hong Li. A seed-driven bottom-up machine learning framework for extracting relations of various complexity. In *Proceedings of ACL07, 584–591*, 2007.

[17] Gong, Jun and Oard, Douglas. Determine the Entity Number in Hierarchical Clustering for Web Personal Name Disambiguation. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

[18] Hall, Mark, Frank, Eibe, Holmes, Geoffrey, Pfahringer, Bernhard, Reutemann, Peter, and H.Witten, Ian. The WEKA Data Mining Software: An Update;. In *SIGKDD Explorations, Volume 11, Issue 1*, 2009.

[19] Han, Xianpei and Zhao, Jun. CASIANED: People Attribute Extraction based on Information Extraction.

[20] Kautz, Henry, Selman, Bart, and Shah, Mehul. The hidden web. *Artificial Intelligence Magazine*, 18(2):27–36, 1997.

[21] Kosala, Raymond and Blockeel, Hendrik. Web Mining Research: A Survey. *SIGKDD Explorations*, 2:1–15, 2000.

[22] Kushmerick, Nicholas. Wrapper Induction: Efficiency and Expressiveness. *Artificial Intelligence*, 118(1-2):15–68, April 2000.

[23] Lafferty, John, McCallum, Andrew, and Pereira, Fernando. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML-01, 18th Int. Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, 2001.

[24] Liu, Bing and Chang, Kevin Chen Chuan. Editorial: special issue on web content mining. *SIGKDD Explor. Newsl.*, 6(2):1–4, December 2004.

[25] McCallum, Andrew. Information Extraction: Distilling Structured Data from Unstructured Text. *Queue*, 3(9):48–57, 2005.

[26] Mobasher, B. Web Usage Mining. In *Encyclopedia of Data Warehousing and Mining*, 2006.

[27] Nagy, István T. and Farkas, Richárd. Person attribute extraction from the textual parts of web pages. In *Third Web People Search Evaluation Forum (WePS-3), CLEF 2010*, 2010.

[28] Pasca, Marius. Outclassing Wikipedia in open-domain information extraction: weakly-supervised acquisition of attributes over conceptual hierarchies. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 639–647, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[29] Sang, Erik F. Tjong Kim and Meulder, Fien De. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of CoNLL-2003*, pages 142–147, 2003.

[30] Sekine, Satoshi and Artiles, Javier. WePS 2 Evaluation Campaign: Overview of the Web People Search Attribute Extraction Task. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

[31] Srivastava, Jaideep and Cooley, Robert. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, 1:12–23, 2000.

[32] Szarvas, György, Farkas, Richárd, Felföldi, László, Kocsor, András, and Csirik, János. A highly accurate Named Entity corpus for Hungarian. In *Proceedings of International Conference on Language Resources and Evaluation*, 2006.

[33] Szarvas, György, Farkas, Richárd, and Kocsor, András. A Multilingual Named Entity Recognition System Using Boosting and C4.5 Decision Tree Learning Algorithms. Discovery Science 2006, 2006.

[34] Toutanova, Kristina, Klein, Dan, Manning, Christopher D., and Singer, Yoram. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL*, pages 252–259, 2003.

[35] Varga, Dániel and Simon, Eszter. Hungarian named entity recognition with a maximum entropy approach. *Acta Cybernetica*, 18(2):293–301, 2007.

[36] Watanabe, Keigo and Bollegala, Danushka. MIVTU: A Two-Step Approach to Extracting Attributes for People on the Web. In *Proceedings 2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

[37] Yi, Lan. Web page cleaning for web mining through feature weighting. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 43–50, 2003.

Received 20th June 2011

Acta Cybernetica **20** (2012) 441–458.

Loop Elimination, a Sound Optimisation Technique for PTTP Related Theorem Proving*

Zsolt Zombori and Péter Szeredi†

Abstract

In this paper we present *loop elimination*, an important optimisation technique for first-order theorem proving based on Prolog technology, such as the Prolog Technology Theorem Prover or the DLog Description Logic Reasoner. Although several loop checking techniques exist for logic programs, to the best of our knowledge, we are the first to examine the interaction of loop checking with ancestor resolution. Our main contribution is a rigorous proof of the soundness of loop elimination.

Keywords: resolution, theorem proving, Prolog, PTTP, loop elimination

1 Introduction

Resolution [8] has long been one of the major approaches to automated theorem proving. Besides its theoretical importance, many academic as well as industrial implementations have been built using resolution. Prolog [7] is a programming language that too implements a resolution based inference mechanism. Prolog is highly optimised and has a very high inference rate, thanks to which more complex reasoning systems, such as the Prolog Technology Theorem Prover (PTTP) [9] and the DLog system [6] have been built on top of Prolog. These systems exploit the backtracking mechanism of Prolog to search for a proof of the initial goal. Efficiency is crucial since these systems typically need to explore a huge search space. In this paper we present an optimisation technique called *loop elimination* for Prolog based reasoning, which can make a tremendous impact on the speed of both of the aforementioned systems. This technique prevents logic programs from trying to prove the same goal over and over again, thus avoiding certain types of infinite loops.

Detecting loops to prune the search space for logic programs is not new, see for example [2]. However, the systems that we are interested in extend standard Prolog execution with a technique called *ancestor resolution*, that corresponds to the positive factoring inference rule. In the presence of ancestor resolution, the considerations that trivially justify

*The work reported in the paper has been developed in the framework of the project "Talent care and cultivation in the scientific workshops of BME" project. This project is supported by the grant TÁMOP - 4.2.2.B-10/1–2010-0009

†Budapest University of Technology and Econcomics, Department of Comptuer Science and Information Theory, Budapest, Magyar tudósok körútja 2. H-1117, Hungary, E-mail: {zombori,szeredi}@cs.bme.hu

loop elimination do not hold. It is easy to see that trying to prove a goal that is identical to some goal that we are already in the process of proving yields no useful solution and the corresponding proof attempt can be aborted. However, it is far from trivial that the same holds in case the two goals are identical only *modulo ancestor list*, i.e., they can be different in one of their arguments, namely in their list of ancestors. Our paper proves this stronger claim. We are not aware of any other work exploring the interaction between loop elimination and ancestor resolution.

In Section 2 we provide an overview of resolution reasoning and Prolog programming, that will be necessary for understanding the rest of the paper. In Section 3 we examine logic programs in terms of termination and identify the sources of infinite execution. Section 4 contains our main contribution: we define loop elimination and prove its soundness. We end the paper with some concluding remarks in Section 5.

2 Background

In this section we provide some background information about first-order resolution (Subsection 2.1) and its connection to the Prolog programming language (Subsection 2.2). In Subsection 2.3 we present the Prolog Technology Theorem Prover (PTTP), a complete first-order reasoner built-on top of Prolog. Finally, in Subsection 2.4, we give an overview of DLog, a Description Logic reasoner that implements a PTTP like approach. We expect the reader to be familiar with the basics of First-Order Logic.

2.1 Resolution Theorem Proving

Resolution [8] is a powerful method for proving first-order theorems. Directly, it is used to check the consistency of a set of first-order clauses, however, all common reasoning tasks – such as entailment analysis – can be easily reduced to consistency check. *Clauses* are first-order formulae satisfying the following properties: all variables are universally quantified, all quantifiers are at the beginning of the formula and the quantifier-free part is a disjunction of *literals*, i.e., possibly negated atomic predicates. It is well known that any set of first-order formulae can be translated into an equisatisfiable set of clauses (for example, see [3]). Since all variables are universally quantified, it is customary to omit the quantifiers. We will do so in the following. Resolution defines two inference rules, called *Binary Resolution* and *Positive Factoring*, presented in Figure 1. In the figure, the clauses above the bar are the premises of the inference and the clause under the bar is the conclusion. σ is the *most general unifier* of B and C, i.e., a variable substitution to terms that satisfies two properties: (1) after the substitution B and C are identical, i.e., $B\sigma = C\sigma$, and (2) σ is a most general substitution that satisfies (1). In Figure 2, we illustrate the

$$\frac{A \vee B \qquad \neg C \vee D}{A\sigma \vee D\sigma} \qquad\qquad \frac{A \vee B \vee C}{A\sigma \vee C\sigma}$$

Figure 1: Binary Resolution and Positive Factoring

application of the the two inference rules. On the left side the Binary Resolution rule is

used and on the right side the Positive Factoring rule fires. The most general unifier is the same in both examples: variable y is mapped to x and every other variable is mapped to itself.

$$\frac{A(x)\vee B(x) \qquad \neg B(y)\vee D(y)}{A(x)\vee D(x)} \qquad\qquad \frac{A(x)\vee B(x)\vee B(y)\vee D(y)}{A(x)\vee B(x)\vee D(x)}$$

Figure 2: Examples illustrating the Binary Resolution and Positive Factoring inference rules

Theorem 1. *Binary Resolution and Positive Factoring yield a calculus that is* sound *and* complete. *This means that a set of clauses is inconsistent if and only if there is a finite series of clauses $C_1, C_2, \ldots, C_n = \square$, where $\square$ denotes the empty clause, such that each clause is either a member of the initial clause set or is obtained as a conclusion of Binary Resolution or Positive Factoring with premises selected from preceding clauses.*

A proof of Theorem 1 can be found, for example, in [8].

Linear resolution As Theorem 1 indicates, resolution captures logical entailment very well. However, finding a deduction of the empty clause to show inconsistency can be rather tedious as we are given no guidance as to what clauses should be resolved in what order. To address this, various selection strategies have been devised, among them *linear resolution.*

Linear resolution is motivated by the idea that if we add a clause to a set of clauses that is considered consistent, then we only have to check the interactions that the new clause can have with the rest. Hence, in the first step, we resolve the new clause with some other, and in all subsequent steps, one of the premises will be the conclusion of the preceding step. Unfortunately, while in linear resolution the number of possible deductions is greatly decreased, we lose completeness. However, linear resolution remains complete for a restricted type of clauses that contain at most one positive literal, called *Horn clauses.* Besides, as it is shown in [5], linear resolution can be extended with a technique called *ancestor resolution* (see below in Subsection 2.3) which yields a complete calculus for the whole of First-Order Logic.

2.2 Programming in Prolog

Prolog [7] is a declarative programming language equipped with a built-in logical inference mechanism that corresponds to linear resolution. This mechanism is complete for Horn clauses, which correspond directly to Prolog rules. A rule has three parts: a head containing the only positive literal, the symbol ':-' and a body which is the list of negative literals without negation, separated by commas. So, for instance, the Horn clause $P(X) \vee \neg Q_1(X) \vee \neg R(X,Y) \vee \neg Q_2(Y)$ corresponds to the Prolog rule

```
P(X) :- Q1(X), R(X,Y), Q2(Y).
```

The semantics of this rule is as follows: if all atoms in the body are true, then so is the atom in the head. A Prolog program is a set of rules that can be used to prove a query atom, called *goal*. The program will try to unify the goal with some rule head, and in case of a successful unification, it will recursively try to prove each statement in the body. If the goal matches more than one rule head, then the program remembers this by creating a so called *choice point* and proceeds with the first matching rule. If we manage to unify the goal with a bodiless rule head, then the goal is proved. If the inference fails, because there is no matching rule head, then we roll back to the last choice point and proceed with the next matching rule. This algorithm corresponds to linear resolution that starts from the negation of the query and that is always resolved in its first literal. This mechanism is very efficient in that it starts out from the goal and examines only those rules that have a potential to answer it.

2.3 Prolog Technology Theorem Proving

The Prolog Technology Theorem Prover approach (PTTP) was developed by Mark E. Stickel in the late 1980's [9]. PTTP is a sound and complete first-order theorem prover, built on top of Prolog. An arbitrary set of general clauses is transformed into a set of Horn-clauses that correspond to Prolog rules. Prolog execution on these rules yields first-order logic reasoning.

In PTTP, to each first-order clause we assign a set of Horn-clauses, the so-called *contrapositives*. The first-order clause $L_1 \vee L_2 \vee L_3 \vee \cdots \vee L_n$ has n contrapositives of the form $L_k \leftarrow \neg L_1, \ldots, \neg L_{k-1}, \neg L_{k+1}, \ldots, \neg L_n$, for each $1 \leq k \leq n$. Having removed double negations, the remaining negations are eliminated by introducing new predicate names for negated literals. For each predicate name P a new predicate name *not_P* is introduced, and all occurrences of $\neg P(X)$ are replaced with *not_P*(X), both in the head and in the body. The link between the separate predicates P and *not_P* is provided using *ancestor resolution*, see below. For example, the clause $A(X) \vee \neg B(X) \vee \neg R(X,Y)$ is translated into three Prolog rules, each with different rule head:

```
A(X)       :-  B(X), R(X,Y).
not_B(X)   :-  not_A(X), R(X,Y).
not_R(X,Y) :-  not_A(X), B(X).
```

Thanks to using contrapositives, each literal of a first-order clause appears in the head of a Horn clause. This ensures that each literal can participate in a resolution step, in spite of the restricted selection rule of Prolog.

Next, let us see how PTTP implements positive factoring. Suppose we want to prove the goal A and during execution we obtain the subgoal $\neg A$. What this means that by this time we have inferred a rule, according to which if a series of goals starting with $\neg A$ is true, then A follows:

$$\texttt{A} \leftarrow \texttt{not\_A}, \texttt{P}_1, \texttt{P2}, \ldots \texttt{P}_k.$$

The logically equivalent first-order clause is

$$A \vee A \vee \neg P_1 \vee \neg P2 \vee \cdots \vee \neg P_k$$

from which we see immediately that the two occurrences of A can be unified, so there is no need to prove the subgoal `not_A`. This step is called *ancestor resolution* [5], which corresponds to the positive factoring inference rule. Ancestor resolution is implemented in Prolog by building an *ancestor list* which contains *open* predicate calls (i.e. goals that we started but have not yet finished proving).

Ancestor resolution is the inference step that checks if the ancestor list contains a goal which can be matched with the negation of the current goal. If this is the case, then the current goal succeeds and the unification with the ancestor element is performed. Note that in order to retain completeness, as an alternative to ancestor resolution, one has to try to prove the current goal using normal resolution, too. This is important if the ancestor element contains variables and a different proof can yield a different variable substitution.

There are two further features in the PTTP approach. First, to avoid infinite loops, iterative deepening is used instead of the standard depth-first Prolog search strategy. Second, in contrast with most Prolog systems, PTTP uses occurs check during unification, i.e., for example terms X and $f(X)$ are not allowed to be unified because this would result in a term of infinite depth.

To sum up, PTTP uses five techniques to build a first-order theorem prover on the top of Prolog: contrapositives, renaming of negated literals, ancestor resolution, iterative deepening, and occurs check.

2.4 DLog, a Description Logic Reasoner

The system DLog [6] is a Description Logic (DL) [1] reasoner for the $\mathcal{SHIQ}$ DL language, geared towards data reasoning, i.e., so called ABox inference. It proceeds by transforming the initial knowledge base into a set of first-order clauses and then performs a two-phase reasoning. The first phase deals only with the so called terminology box (TBox) part of the knowledge base that contains general background knowledge. The point of this phase is that by the end we obtain a syntactically simpler set of clauses to be used in the subsequent data reasoning. The second phase uses Prolog to perform the rest of the reasoning in a way similar to PTTP. Due to the syntactically simpler input clause set, the general PTTP approach can be optimised and simplified in a number of ways.

In [10] we describe the algorithm of the first phase, as a result of which we obtain clauses of the following types:

$$\neg R(x,y) \vee S(y,x) \tag{1}$$

$$\neg R(x,y) \vee S(x,y) \tag{2}$$

$$\mathbf{P}(x) \tag{3}$$

$$\bigvee_i \left(\mathbf{P_1}(x_i) \vee \left(\bigvee_j \neg R(x_i,y_j) \vee \bigvee_j \mathbf{P_2}(y_j) \vee \bigvee_{j_1,j_2} (y_{j_1} = y_{j_2}) \right) \right) \tag{4}$$

$$R(a,b) \tag{5}$$

$$C(a) \tag{6}$$

$$a = b \tag{7}$$

$$a \neq b \tag{8}$$

where $\mathbf{P}(x)$ is a shorthand for $(\neg)P_1(x) \vee (\neg)P_2(x) \vee \cdots \vee (\neg)P_k(x)$. Note some nice properties of our clause set:

1. There are no function symbols.
2. In the contrapositives generated from these clauses a negative binary literal can only appear in the body in case the head is a negative binary literal (clauses of type 1 and 2).
3. For every clause that contains binary literals, all variables occur in some binary literal.
4. Clauses that do not contain binary literals have at most one variable.

As we will see later, these nice properties allow for specializing PTTP to obtain a terminating decision procedure for the $\mathcal{SHIQ}$ DL language.

3 Termination of Logic Programs

Given that first-order logic is undecidable, it is not surprising that the Prolog Technology Theorem Prover is not guaranteed to terminate. In this section we review the ways in which a logic program can fall short of termination. Afterwards, we compare PTTP and DLog with respect to termination.

3.1 Sources of infinite execution

We identify three sources of infinite execution:

- If the program contains **function symbols**, then we might obtain terms of ever increasing depth. Consider, for example, the following simple program:

  ```
  p(X)      :-  p(f(X)).
  ```

  If we attempt to prove $p(a)$ using the above rule, we will end up reducing it to the proof of $p(f(a))$, $p(f(f(a)))$ etc. and the program will never stop.

- A proof attempt might visit infinitely many goals if an unbounded number of **new variables** can be introduced during the proof. This is the case for example with the transitivity rule:

  ```
  r(X,Y)    :-  r(X,Z), r(Z,Y).
  ```

  It is easy to see that a proof attempt for the goal $r(a,b)$ using the above rule will generate infinitely many $r(X,Y)$ subgoals, always with fresh variables.

- Even if both the depth of terms and the number of variables can be bounded, the program might fall into a **loop** and attempt to prove the same goal over and over again. For example, the program consisting of the following rule

```
p(X)        :-  p(X).
```

will never terminate, even though there are no function symbols and no new variables are introduced.

One can see easily that the above list is exhaustive. If the number of variables is bounded and there are no functions, then the total set of terms is that of the variables and the constants appearing in the program, i.e., it is finite. Since the predicate names are also finite, there can be finitely many different goals. If there are no loops, even if a proof attempt goes though all possible goals (the worst case), it will eventually terminate.

Hence, we conclude that infinite execution is due exactly to three aspects of logic progams: function symbols, the proliferation of new variables and loops.

3.2 Termination in DLog

In light of the preceding subsection, let us reexamine the input clause set of the DLog data reasoner. We see immediately that the absence of function symbols eliminates one of the three sources of infinite execution.

We shall see that new variables are not introduced, either. The second nice property of the input clause set is that the resulting contrapositives only contain a negative binary literal in the body in case the head is a negative binary literal. This means that we can only encounter negative binary subgoals if the initial query itself is a negative binary goal. In $\mathcal{SHIQ}$ DL reasoning, however, negative binary queries are forbidden, so all contrapositives with a negative binary literal are unnecessary and can be disposed of. Consequently, in our logic program binary literals will only appear positively. For proving such binary goals only contrapositives from clauses of type 1 and 2 are available:

```
r(X,Y)      :-  s(X,Y).
r(X,Y)      :-  s(Y,X).
```

These rules do not introduce new variables. A proof of a binary goal consists of applying such rules possibly several times, until finally we obtain a matching data assertion `r(a,b)`, thanks to which the variables in the binary goal get instantiated. We know that in all rule bodies that contain binary literals every variable occurs in some binary literal (the third nice property of our input clause set). These are the rules that introduce new variables. If, however, we move the binary literals to the front of the body, i.e., we prove the binary goals first, by the time we reach the unary goals, they become ground. Hence, any unary goal in the body either contains the same variable as the one in the head – in case the rule contains no binary predicates – or else it is ground by the time it is called. New variables may appear only for a short time – until we prove the binary goals holding them. Hence, DLog will never encounter infinitely many new variables during a proof attempt.

If there are no terms of increasing depth and variables do not proliferate, then the only way a DLog program may not terminate is if it falls in an infinite loop and proves the same goal repeatedly.

3.3 Eliminating Loops

We have seen that there are three independent features that can make a PTTP execution non-terminating, of which only one, namely loops can occur in DLog programs. In Section 4 we shall show that proofs containing such loops are not necessary for completeness. This result yields an important optimization for both PTTP and DLog, called *loop elimination*. General PTTP still has to cope with infinite proof attempts (due to the other two sources) and hence has to use iterative deepening, i.e., build several proof attempts in parallel. However, even if loop elimination does not allow for changing the proof search strategy, but it still prunes the search space significantly. In DLog, loop elimination eliminates the only remaining source of infinite proofs. Accordingly, DLog always terminates and uses the standard depth-first search strategy of Prolog, which gives much better performance than iterative deepening.

4 Loop Elimination

In this section we present the optimization heuristic *loop elimination* for both PTTP and DLog. In the literature, loop elimination is often referred to as *identical ancestor pruning*, see for example [9] or [4]. Although both systems employ this optimisation, there has not yet been any rigorous proof of its soundness. In Subsection 4.1 we describe *proof trees* that can be used to represent Prolog execution. Afterwards, Subsection 4.2 contains the proof of soundness.

Definition 1 (Loop elimination). *Let P be a Prolog program and G a Prolog goal. Executing G w.r.t. P using* loop elimination *means the Prolog execution of G extended in the following way: we stop the given execution branch with a failure whenever we encounter a goal H that is identical to an open subgoal (that we started, but have not yet finished proving). Two goals are identical only if they are syntactically the same.*

Loop elimination is very intuitive. If, for example, we want to prove goal G and at some point we realise that this involves proving the same goal G, then there is no point in going further, because 1) either we fall in an infinite loop and obtain no proof or 2) we manage to prove the second occurrence of G in some other way that can be directly used to prove the first occurrence of the goal G. This is the standard justification that we find in the literature. For example [4] says:

> Identical ancestor pruning (IAP) is a powerful pruning heuristic in a model elimination search. Imagine, in the course of expanding a ME proof space for a particular goal P, that one were to encounter that same goal P again. One of two situations must hold:
>
> 1. There are no proofs of P from this database (because it doesn't logically follow).
> 2. Whether or not there is a proof using this second occurrence of P, there must be another proof of the original P not using it. Also, the different proof occurs at a shallower depth.

> This is true because the second occurrence must eventually be proven somehow, so this recursion must bottom out. And then, by whatever proof this second occurrence succeeds, an analogous proof path must exist below the first occurrence of P. In either case, it is justifiable to prune the space below the second occurrence of P.

Things get complicated, however, due to ancestor resolution. The two G goals have different ancestor lists and it can be the case that we only manage to prove the second G due to the ancestors that the first G does not have. As it will turn out in the rest of this section, while we can indeed construct a proof of the first G from that of the second, this proof might have to be very different from the original one.

4.1 Proof Trees

In this subsection we introduce *proof trees*, that are used to represent Prolog execution. We will only consider trees in the context of a PTTP like Prolog program, more precisely we will assume that the program contains all contrapositives. Each tree node has a unique name and is labelled with a goal: `(Name:Goal)` refers to a node called `Name` and labelled with goal `Goal`. The root is labelled with the initial goal to be proved. Suppose the current goal G is unified with the head of rule

$$\texttt{G:-}\ \ \texttt{B}_1,\texttt{B}_2,\ldots,\texttt{B}_k.$$

In this case, the node labelled G will have k children, each labelled $B_1, B_2, \ldots, B_k$, respectively. In each inference step, the validity of a goal is reduced to the validity of a set of goals in the children. After a successful execution, we obtain a proof tree such that each of its leaves can be considered true without further proof. We formalise this in the following definitions.

Definition 2. *An* atomic proof tree *consists of a root node labelled $A\sigma$ with children labelled $B_1\sigma, B_2\sigma, \ldots, B_n\sigma$, where σ is a variable substitution. We say that the atomic proof tree is* valid *if the corresponding Prolog program contains a rule*

$$\texttt{A:-}\ \ \texttt{B}_1,\texttt{B}_2,\ldots,\texttt{B}_n.$$

A valid atomic proof tree can be seen as an instance of a rule. A proof tree *is built from atomic proof trees by matching nodes of identical labels. A proof tree is* valid *if all constituting atomic proof trees are valid.*

Remark 1. The labels of proof trees are atomic predicates that can contain variables. Note that labels $p(X)$ and $p(Y)$ are not identical.

Definition 3. *In a valid proof tree, a node labelled A is called* complete *if either 1) A can be unified with the head of a bodiless Prolog rule or 2) the node has an ancestor labelled $\neg A$ (ancestor resolution). A valid proof tree is complete if all its leafs are complete.*

To each successful Prolog execution that employs ancestor resolution, we can assign a complete proof tree.[1] In fact, the execution mechanism can be seen as a search in the space of complete proof trees. While standard Prolog will not necessarily traverse the whole space (because it might fall into an infinite loop), both PTTP and DLog are built so that they can enumerate all complete proof trees. This means that it is enough to show the existence of a complete proof tree to guarantee a successful PTTP or DLog execution.

Definition 4. *For an arbitrary child b of an atomic proof tree, the transformation* flipping over along the *b* child *is defined as follows: the root node is switched with its child b and their labels are negated. The rest of the tree is unaltered. This transformation is illustrated in Figure 3.*

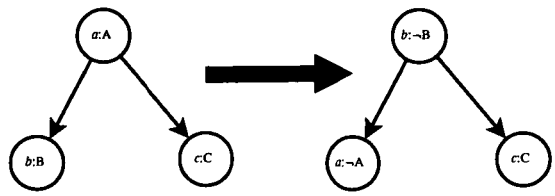

Figure 3: Flipping over along the *b* child

Lemma 1. *For every valid atomic proof tree, the atomic tree obtained after flipping over along a child results in a valid atomic proof tree.*

Proof. Let T be an atomic proof tree with the root node labelled $A\sigma$ and children labelled $B\sigma, C_1\sigma, \ldots, C_k\sigma$. T is an instance of the Prolog clause

$$\mathtt{A} :- \ \mathtt{B}, \mathtt{C}_1, \ldots, \mathtt{C}_k.$$

which is a contrapositive of the first-order clause $A \vee \neg B \vee \neg C_1 \vee \cdots \vee C_k$. Since the Prolog program contains all contrapositives of this clause, we also have

$$\mathtt{not\_B} :- \ \mathtt{not\_A}, \mathtt{C}_1, \ldots, \mathtt{C}_k.$$

an instance of which corresponds to the flipped over version of T. □

Note that flipping over allows us to move between contrapositives of the same first-order clause.

Definition 5. *The transformation* flipping over along the $a, \bar{a}$ branch *is defined on proof trees as follows: let F be a proof tree, with a node* $(a : A)$ *which has a leaf descendant* $(\bar{a} : \neg A)$. *The nodes on the path from a to* $\bar{a}$ *are* $a = x_0, x_1, \ldots, x_{n-1}, x_n = \bar{a}$. *To this tree we assign a tree* F' *which differs from F only in the subtree rooted at a. This subtree contains a branch* $y_0 = x_n, y_1 = x_{n-1}, \ldots, y_i = x_{n-i}, \ldots, y_n = x_0$, *and the label of each of these nodes is negated. Furthermore, each* y_i *in* F' *has the same siblings as* x_{n-i+1} *in F. The subtrees under the siblings are left unaltered. This transformation is illustrated in Figure 4.*

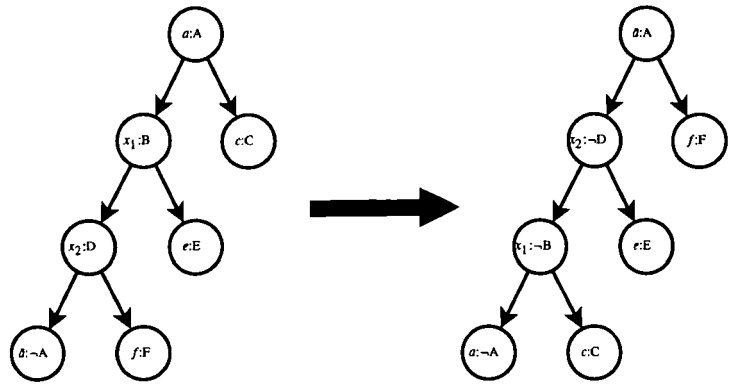

Figure 4: Flipping over along the $(a, \bar{a})$ branch

Lemma 2. *If we have a complete proof tree T that contains nodes $(a : A)$ and $(\bar{a} : \neg A)$ such that $\bar{a}$ is a leaf descendant of a, then the tree obtained after flipping T along the $(a, \bar{a})$ branch is a valid proof tree.*

Proof. The new downward path $\bar{a} \rightarrow a$ consists of atomic trees that are the flipped over versions of the atomic trees of the initial upward path $\bar{a} \rightarrow a$. We know from Lemma 1 that flipping over a valid atomic proof tree yields another valid atomic proof tree, hence the whole new proof tree is valid. □

Remark 2. Although we obtained a valid proof tree after flipping over, the proof tree is not necessarily complete. This is because some ancestor lists change and branches that previously terminated in ancestor resolution might have to be expanded further (because the required ancestor disappeared).

4.2 The Soundness of Loop Elimination

In this subsection we show that for every complete proof tree that contains loops, one can construct a complete proof tree that is loop free.

Definition 6. *A complete proof tree is said to contain a loop L if it contains a pair of nodes $(p_1 : P), (p_2 : P)$, for some label P, such that p_2 is a descendant of p_1. Node p_1 is called the* top node *and node p_2 the* bottom node *of the loop L. We define the* depth *of L to be the distance of p_1 from the root.*

Definition 7. *A node $n : N$ is said to be* eligible for ancestor resolution *if it has an ancestor with label $\neg N$. If an inner node is eligible for ancestor resolution, then it is called a* bad node.

Bad nodes are called bad, because they are unnecessarily expanded. There is no need to provide a proof tree under a bad node, since it is complete even if it remains a leaf.

[1] In the Logic Programming community, it is customary to reserve the name proof tree only for complete proof trees. We introduce the notion of completeness because we will have to refer to trees that are not fully expanded.

Lemma 3. *If we have a complete proof tree that contains a bad node n, then the tree obtained after removing the subtree under n yields a complete proof tree in which n is not bad any more.*

Proof. Removing the subtree under n makes n a leaf node. However, n is complete due to ancestor resolution. The rest of the leaves are unaltered, so they remain complete. Hence, the new proof tree is complete. □

Definition 8. *We define the* loop-depth *of a tree T with a pair of integers* $(-D,C)$*, where D is the minimum depth of all loops in T and C is the number of nodes that are bottom nodes of some loop of depth D. If the tree contains no loops, then its loop-depth is* $(-\infty,0)$*. Loop-depths are comparable using lexicographic ordering, i.e., loop-depth* (A,B) *is less than loop-depth* (C,D) *if and only if either* $A < C$ *or else* $A = C$ *and* $B < D$*.*

Lemma 4. *Let F be a complete proof tree with loop-depth LD that contains at least one loop. It is possible to find another complete proof tree* F' *for the same goal (i.e., with the same label in the root) such that the loop-depth of* F' *is strictly less than LD.*

Proof. The loop-depth of F is LD=$(-D,C)$. This means that there is at least one loop of depth D and there are no loops with depth less than D. Let L be one such loop with top and bottom nodes $(p_1 : P)$ and $(p_2 : P)$, respectively. First, we eliminate all bad nodes by removing the subtrees rooted at the bad nodes. According to Lemma 3, we obtain a complete proof tree.

In case the elimination of the subtrees under bad nodes eliminates loop L, then the obtained complete proof tree has loop-depth $(-D_2,C_2)$. In case there were no other loops of depth D in F then $D_2 > D$. Otherwise, $D_2 = D$ and $C_2 = C - 1$. In either case $(-D_2,C_2) < (-D,C)$, so our lemma is satisfied.

Otherwise, in the obtained tree, all nodes that are eligible for ancestor resolution are leaf nodes. The ancestor list of p_2 contains the ancestors of p_1 plus the nodes on the path between p_1 and p_2. Let *ANC* denote the set of nodes between p_1 and p_2.

In case none of the nodes in *ANC* play any role in the proof of p_2 (i.e., they do not participate in ancestor resolution), the proof of p_1 can be directly replaced with that of p_2, eliminating loop L. This is illustrated in Figure 5. We obtained a complete proof tree F' with loop-depth $(-D_2,C_2)$. In case there were no other loops of depth D in F then $D_2 > D$. Otherwise, $D_2 = D$ and $C_2 = C - 1$. In either case $(-D_2,C_2) < (-D,C)$, so our lemma is satisfied.

The situation is more complicated when some nodes in *ANC* participate in ancestor resolution under p_2. Among these, let $(a : A)$ be the lowest one (i.e., the last one to enter the ancestor list). Somewhere under p_2 there is a leaf $(\bar{a} : \neg A)$ that is complete due to ancestor resolution. Let us flip over F along the branch $(a,\bar{a})$. In the flipped over branch the nodes between a and $\bar{a}$ will appear with negated labels and in inverse order. Afterwards, we once more eliminate all bad nodes by removing the subtrees under them. Node p_2 is on the path between a and $\bar{a}$, so its label will turn to $\neg P$, which makes p_2 eligible for ancestor resolution. Hence, when we eliminate badness, we eliminate the subtree under p_2. As a result, loop L disappears. An example of this is shown in Figure 6. We know that flipping a complete proof tree results in a valid proof tree, but it is not necessarily

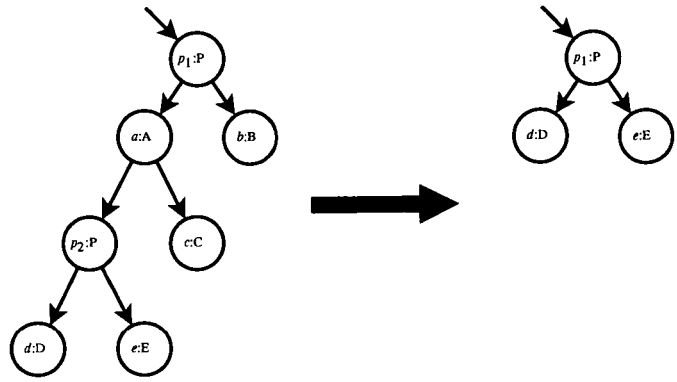

Figure 5: Replacing the proof of p_1 with that of p_2

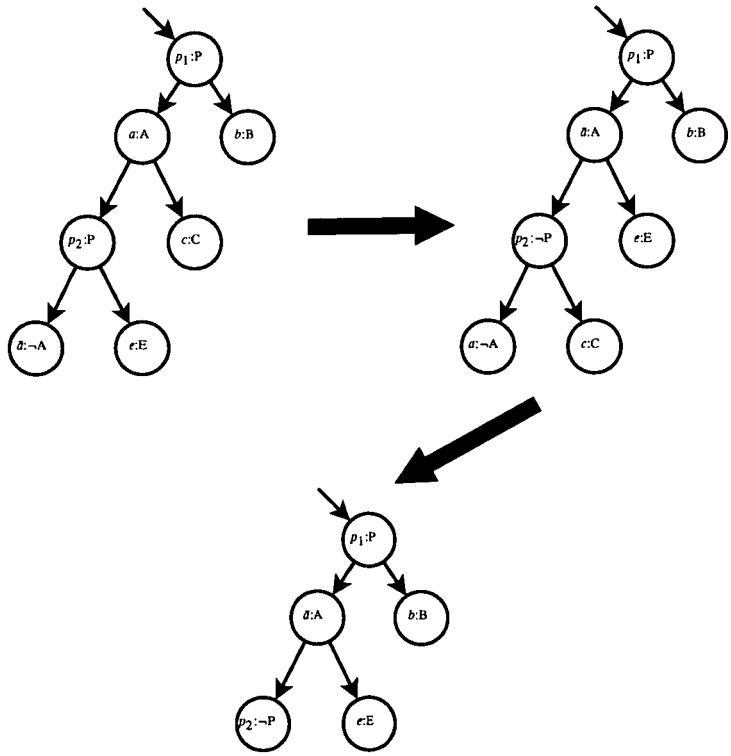

Figure 6: Flipping over along the $(a,\bar{a})$ branch, then bad node elimination

complete, because some goals that previously succeeded with ancestor resolution might loose the required ancestor (cf. Remark 2). This is the case when there is a node $(b : B)$ under a and somewhere underneath there is a leaf $(\bar{b} : \neg B)$. Node b has to be on the path between a and $\bar{a}$ otherwise b will continue to be an ancestor of $\bar{b}$ and their labels will not change. There are two possibilities:

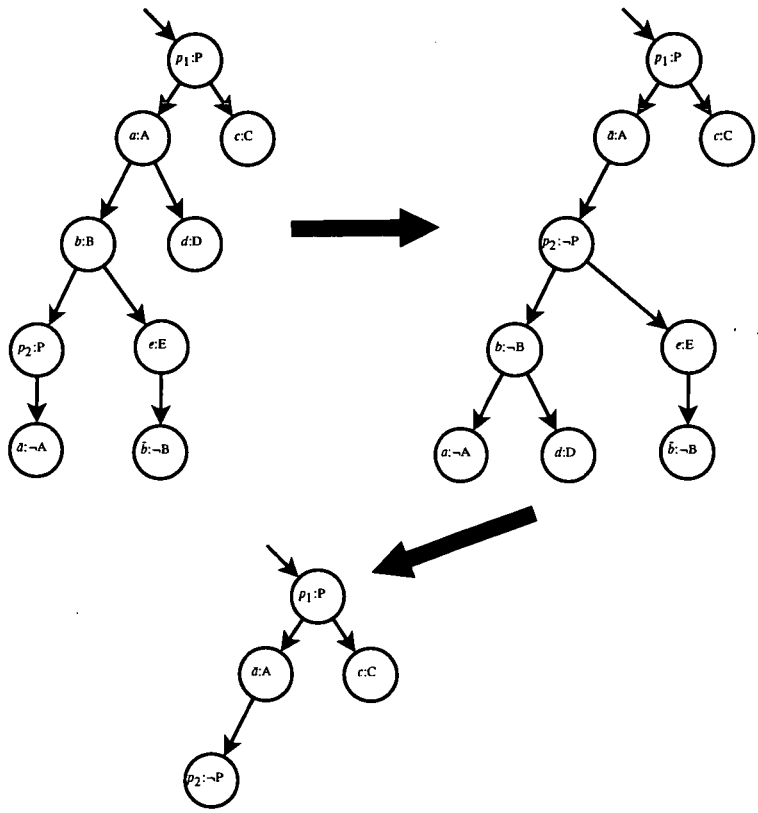

Figure 7: Ancestor resolution eliminates both b and $\bar{b}$

1. As it is illustrated in Figure 7, b lies between a and p_2. Then, $\bar{b}$ cannot appear under p_2, because a was chosen to be the lowest node participating in ancestor resolution under p_2. Hence, $\bar{b}$ appears under b, but not under p_2. After flipping, both b and $\bar{b}$ will appear under p_2, so they will be eliminated when we eliminate the badness of p_2. Hence, this case will not yield any incomplete leaves.

2. We illustrate the second case, namely when b is under p_2 in Figure 8. We will treat all such nodes together, i.e., let $(b_1 : B_1), (b_2 : B_2), \ldots (b_k : B_k)$ be nodes on the path between a and $\bar{a}$ (nodes b, c on Figure 8), such that each b_i has at least one leaf descendant $(\bar{b}_{il} : \neg B_i)$. The nodes are ordered so that b_1 is the closest to p_2 and b_k is the farthest. After flipping over, the labels of these nodes will be negated, i.e., turn to

$\neg B_i$, respectively, and they will appear on the branch leading to p_2 in inverted order, i.e., b_k will be the topmost, while b_1 the lowest.

Let us consider b_1. Due to flipping over, it will lose all its previous descendants. Its new descendants will be its previous ancestors on the path between p_2 and b_1 along with their descendants towards other branches. We claim that none of the new descendants of b_1 can have lost an ancestor which previously allowed for ancestor resolution, i.e., none can be one of $\bar{b}_{il}$. This is because the lost ancestor would have been above b_1, however, b_1 was chosen to be the topmost one. Consequently, the subtree under b_1 after flipping has no incomplete leaves. This subtree in itself is not necessarily complete, because the ancestors of $\bar{a}$ might be needed for some ancestor resolution steps. We express this by saying that the subtree under b_1 is *complete in the context of the ancestors of* $\bar{a}$. In the following, we will always assume the same context (the ancestors of $\bar{a}$) and will omit specifying it whenever it leads to no misunderstanding. The label of b_1 is $\neg B_1$, so we have a complete proof for $\neg B_1$ (again in the context of the ancestors of $\bar{a}$). This means that we can copy the subtree under b_1 to any node $(\bar{b}_{1l} : \neg B_1)$, thus compensating such nodes for the lost ancestor. Note that we need to rename the copied nodes to ensure that each node has a unique name.

We next turn to b_2. Through analogous reasoning we can see that the new leaf descendants of b_2 are either complete or else are incomplete because they lost an ancestor labelled $\neg B_1$. However, by copying the subtree under b_1, we have already turned such leaves into complete trees. Hence, we have a complete proof tree under b_2 (in the context of $\bar{a}$), proving $\neg B_2$, which we copy to any incomplete leaf $(\bar{b}_{2l} : \neg B_2)$ (again assigning new names to the newly created nodes).

We continue the process. In the i^{th} step, we have a complete proof tree under b_i which we copy to any leaf $(\bar{b}_{il} : \neg B_i)$. By the end of the k^{th} step, we obtain a complete proof tree. Note that we make exactly one copying for each leaf $\bar{b}_{il}$ that lost its completeness after flipping over, so copying terminates.

We now obtained a new proof tree F'. Let us show that F' has the properties claimed by the lemma being proved. Flipping over turns the label of p_2 from P to $\neg P$, which makes loop L disappear. New loops can arise (some nodes were negated), however, no such loop can start above or at p_1. We show this by contradiction. Suppose a node $(n_1 : N)$ above or at p_1 obtains a descendant $(n_2 : N)$ after flipping. The labels of the nodes under n_1 in the new tree are either the same or the negated labels that appeared under n_1 before flipping. So, if a new loop appeared, it was either because the bottom node of an already existing loop L_2 was copied or because the label of a descendant of n_1, namely of n_2, changed from $\neg N$ to N. In the first case, the depth of loop L_2 is smaller than the depth of loop L, which is impossible because L was chosen to be a loop of minimum depth (cf. Definition 8. of loop-depth). In the second case, before flipping over, n_2 was eligible for ancestor resolution. Since we eliminated all bad nodes, n_2 was a leaf. However, flipping over does not negate the labels of leaf nodes, so we obtained a contradiction.

We conclude that the possibly arising loops are all of greater depth than the eliminated

loop. Hence, the number of loops of depth D is reduced by one, i.e., the loop-depth of the new tree is strictly less than that of the original tree. □

Theorem 2. *For every complete proof tree containing loops there is a complete proof tree that is loop free.*

Proof. Using the transformation described in Lemma 4, we can create a series of proof trees of the same goal such that the loop-depth is always decreasing. The second component of the loop-depth is a positive integer (the number of loops at minimum depth) which cannot decrease infinitely, so eventually the first component will decrease as well. This means that the minimum depth of the loops increases, i.e. loops get deeper and deeper. There are two possibilities:

1. Eventually, we manage to eliminate each loop after a finite number of iterations. The resulting proof tree satisfies our theorem.
2. The elimination never terminates. Since the loops are getting farther from the root, it follows that the part of the proof tree that is loop free grows beyond any limit. Suppose the initial tree contains n distinct labels in its nodes. The transformation steps involve flipping over, copying subtrees and eliminating nodes, each of which either preserves node labels or introduces the negation of some label to a node. Hence, there can be at most $2n$ distinct labels, i.e., any loop free path from the root node can be at most $2n$ long. This contradicts the assumption that the loop free part of the tree grows beyond any limit. Hence, all loops have to disappear after finitely many iterations.

□

5 Conclusion

Prolog based inference systems like PTTP and DLog can be used to prove a query goal. We have shown is Section 4 that these systems need not explore proof trees that contain loops, because in case there is a complete proof tree, there is one without loops (Theorem 2). This allows for reducing the search space, making both systems faster. Besides, loop elimination is sufficient to make the DLog reasoner terminating, thus allowing one to replace iterative deepening search with depth-first search, which further increases performance.

References

[1] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. F., editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2004.

[2] Bol, Roland N., Apt, Krzysztof R., and Klop, Jan Willem. An analysis of loop checking mechanisms for logic programs. *Theor. Comput. Sci.*, 86:35–79, August 1991.

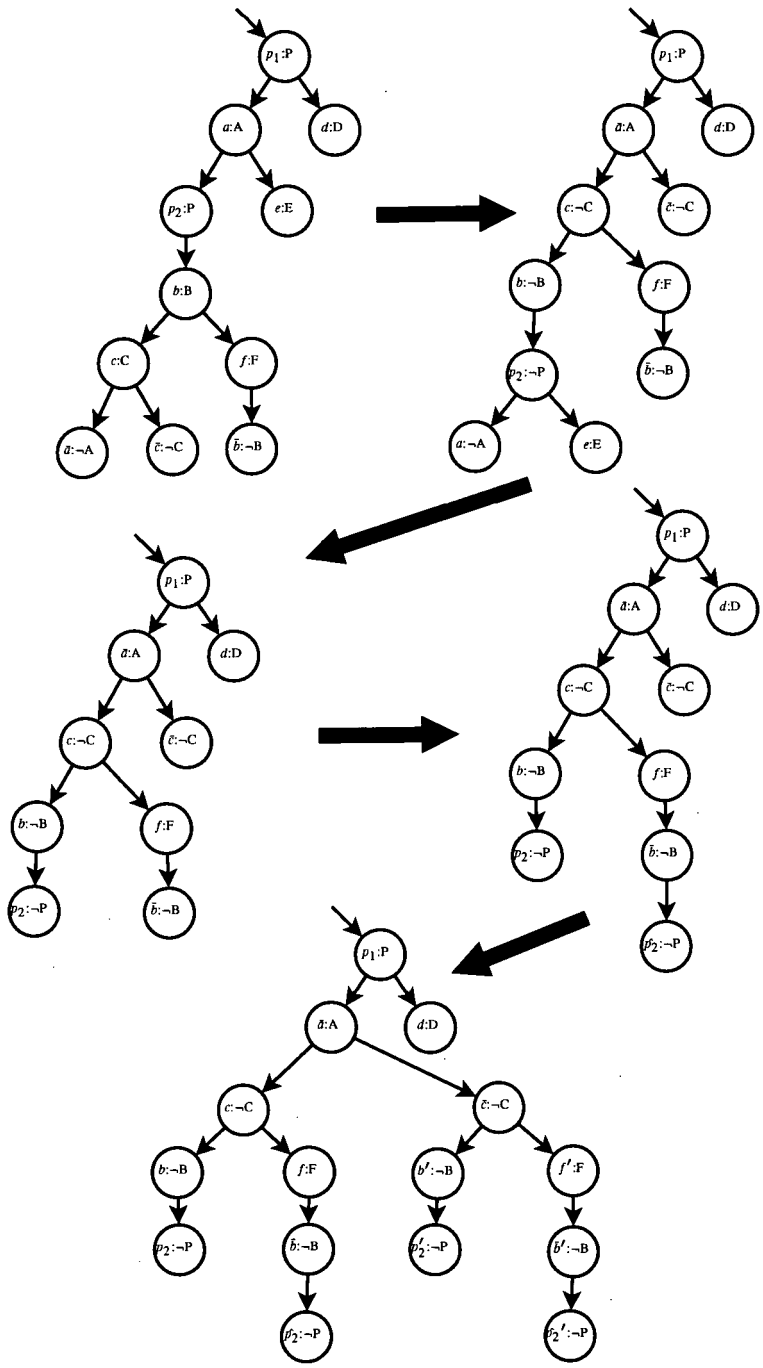

Figure 8: Copying makes first $\bar{b}$, then $\bar{c}$ complete

[3] Fitting, Melvin. *First-order logic and automated theorem proving*. Springer-Verlag New York, Inc., New York, NY, USA, 1990.

[4] Geddis, Donald F. *Caching and non-Horn Inference in Model Elimination Theorem Provers*. PhD thesis, Stanford University, USA, June 1995.

[5] Kowalski, R. and Kuehner, D. Linear resolution with selection function. *Artificial Intelligence*, 2:227–260, 1971.

[6] Lukácsy, Gergely and Szeredi, Péter. Efficient Description Logic reasoning in Prolog: The DLog system. *Theory and Practice of Logic Programming*, 9(03):343–414, 2009.

[7] ISO Prolog standard, 1995. ISO/IEC 13211-1.

[8] Robinson, J. A. A machine-oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, 1965.

[9] Stickel, Mark E. A Prolog technology theorem prover: a new exposition and implementation in Prolog. *Theoretical Computer Science*, 104(1):109–128, 1992.

[10] Zombori, Zsolt. Efficient two-phase data reasoning for Description Logics. In *IFIP AI*, pages 393–402, 2008.

Received 15th November 2011

CONTENTS

ISSN 0324—721 X

Felelős szerkesztő és kiadó: Csirik János
Printed by: Gold Press Nyomda Kft. Szeged